AN ADAPTIVE MESHFREE SOLVER FOR INCOMPRESSIBLE FLOWS

A Dissertation
Presented to
The Academic Faculty

By

Yaroslav V. Vasyliv

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology

May 2020

AN ADAPTIVE MESHFREE SOLVER FOR INCOMPRESSIBLE FLOWS

Approved by:

Dr. Alexander Alexeev, Advisor
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Satish Kumar
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. David Hu
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Wenxiao Pan
Department of Mechanical Engineering
University Wisconsin-Madison

Dr. Igor Pivkin
Institute of Computational Science, Faculty of Informatics
Universita della Svizzera italiana

Date Approved: April 8, 2020

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my family. Without their unconditional support, finishing my graduate studies would not have been possible. To my parents, I will always be grateful for the example you have set, your sacrifice and your bold decision to restart life in America. To my wife, who I would not have met in graduate school if she did not make a similar journey all by herself, thank you for your love, patience, and understanding throughout the highs and lows of our graduate studies.

Second, I would like to acknowledge my Ph.D. advisor, Dr. Alexeev. After several initial semesters at Georgia Tech, I somehow landed a NSF fellowship. The NSF fellowship gave me an opportunity to switch labs and take a gamble on meshfree methods. When I switched labs, Dr. Alexeev also took a gamble and accepted me into his lab. Throughout my graduate studies he has provided resources, support, and encouragement whenever it was needed. While not highlighted in this thesis, he also encouraged broadening my numerical background to include the lattice Boltzmann method.

Last but not least, I would also like to thank the other members of the reading commitee for their valuable time and feedback that they have provided on this dissertation. Additionally, financial support provided from the National Science Foundation (NSF) Graduate Research Fellowship, Grant No. DGE-1148903 is gratefully acknowledged.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Meshfree methods are particularly suitable for problems involving moving interfaces and adaptive refinement as potentially expensive re-meshing is avoided. Despite this advantage, in the context of incompressible flow, meshfree methods are still plagued by a number of issues including grid degradation accompanying large fluid deformations, large stencil requirements, lack of consistency and conservation, restrictive time steps due to explicit weakly compressible fluid models used, and inability to resolve small spatial scales due to grid uniformity. To address a number of these limitations, we develop a consistent, semi-implicit, adaptive meshfree solver based on the arbitrary Lagrangian-Eulerian (ALE) formulation of the incompressible Navier-Stokes equations. The utility of this meshfree ALE approach will be demonstrated by simulating viscous incompressible flow phenomena around complex moving interfaces through a range of Reynolds numbers.

CHAPTER 1
INTRODUCTION

In meshfree methods the governing partial differential equations are discretized in a given computational domain using only a set of points. A mesh comprised of a collection of elements (volumes) with defined faces is not utilized. Instead, elements or volumes are replaced by point clouds. The grid connectivity is then established using a dynamic or static search radius. Differences regarding how the residual is weighted, what shape functions are used, and whether the grid is fixed or moving, lead to different meshfree methods [1–13].

The primary advantage of eliminating the mesh is that problems with moving interfaces and problems requiring adaptive refinement can readily be handled as (potentially) expensive re-meshing is no longer required. Consequently, meshfree methods have had success in a wide array of applications including free surface flows [14–16], multiphase flows [17–21], fracture growth [22, 23] and fluid-structure interaction [6, 7, 24, 25].

Despite their success, the primary disadvantage of eliminating the mesh has loomed in the background. By removing the mesh, it is no longer trivial to obtain a spatial discretization that simultaneously enforces conservation and consistency. More specifically, consistency requires that the stencil coefficients enforce polynomial reproducing conditions [2, 8], whereas conservation requires that the stencil coefficients obey the reciprocity conditions [26]. Separately, both of these conditions can be enforced in a local manner; however, when both conditions are to be enforced simultaneously, a global optimization problem arises for the virtual face areas and volumes [3, 26].

Such a conservative and consistent meshfree approach was first developed by Diyankov in the Uncertain Grid Method (UGM) [3], wherein a linear programming (LP) problem is solved to obtain the stencil coefficients. Diyankov demonstrated the approach qualitatively on the 2D Richtmayer - Meshkov instability and quantitatively on the 2D Poisson equation.

Following a similar vein, Chiu [26] proposed a quadratic programming (QP) problem for the virtual face areas and volumes. Stabilizing the approach with an artificial dissipation scheme, Chiu's approach sharply captures the shocks of the 2D inviscid, compressible flow over a NACA airfoil using a clustered, fixed meshfree grid. Despite the difficulty of resolving the discontinuity and despite abandoning a physical mesh, their results showed good agreement for the pressure coefficient over the surface.

More recently, Trask et al. [27] have developed an approach wherein several independent graph Laplacian problems are solved to obtain the coefficients. The advantage of their approach is that the graph Laplacian problems can be solved in a scalable manner using algebraic multigrid. They demonstrated their virtual finite volume scheme on a 2D Darcy problem with discontinuous coefficients and on the 2D scalar advection-diffusion equation, reporting first order accuracy.

To date, we are not aware of any extension of the above conservative and consistent schemes to viscous incompressible flows; however, we are optimistic such extensions will appear in the near future. We do note that for applications with moving grids, a global problem(s) will need to be solved for the coefficients every time step prior to solving the corresponding sparse linear systems. This dilemma brings up the following fundamental question. For such applications, will it be cheaper to locally (globally) re-mesh and apply existing mesh-based discretizations [28, 29] or will it be cheaper to forgo the physical mesh and solve for coefficients that correspond to a moving virtual mesh?

This challenging question will likely dominate future discussions but will not be addressed in this thesis. Instead, this thesis focuses on addressing several limitations of collocated meshfree solvers as they pertain to viscous, incompressible flow simulations on moving adaptive grids. We will similarly forgo the mesh entirely (i.e., no background mesh); however, when faced with the dilemma of enforcing both conservation and consistency, we instead compromise and choose to only enforce consistency. This tradeoff alleviates the cost of computing stencil coefficients on moving meshfree grids by replac-

ing the global computation with a local one. This compromise is not new and has often been made and applied successfully despite the apparent loss of conservation. Before we state the objectives of this thesis, we first present an overview of meshfree methods with particular attention paid to challenges in collocated meshfree methods.

1.1 Overview of meshfree methods

1.1.1 Smoothed Particle Hydrodynamics

In 1977, Smoothed Particle Hydrodynamics (SPH), arguably the most widespread meshfree approach, was developed for astrophysics applications [30]. In SPH, a continuous kernel approximation is discretized and is used in turn to typically discretize the strong form of the PDE. In its most basic form, SPH results in shape functions equivalent to the kernel (weight or smoothing) function weighted by the volume associated with the particle [2]. The resulting shape functions do not enforce the polynomial reproducing conditions. Consequently, since its extension to other areas of hydrodynamics, SPH has been plagued with consistency issues particularly noticeable near the domain boundaries. Various corrections have been proposed over the years (see e.g., the reviews [11, 31]); however, they have typically fallen short when treating the second derivatives in the viscous term [32]. Moreover, the first consequence of applying these local corrections, is that SPH no longer results in pair-wise forces.

SPH also requires large smoothing lengths h with a correspondingly large number of nearest neighbors ranging between approximately $n = 40$ to $n = 60$ in 2D. For instance, an early work by Morris et al. [33] recommended a smoothing length of $h = 1.5\Delta x$ which for the quintic weight function (compact support $3h$) leads to 69 neighbors on a uniform lattice. In more recent works, Lee et al. [34] used $h = 1.3\Delta x$ and Hu et al. [35] used $h = 1.25\Delta x$ which results in 45 nearest neighbors on a uniform 2D lattice. In 2D, SPH requires 6 to 12 times the number of nearest neighbors of a finite difference (FD) Laplacian stencil. In 3D, the situation is even worse, e.g., in floating body simulations [16] a more

efficient cubic weight function was used (compact support of $2h$) with smoothing length $h = 1.3\Delta x$ resulting in 81 nearest neighbors on a uniform lattice.

Such a large memory bandwidth will severely impact the performance of iterative solvers. To no surprise, most SPH implementations avoid implicit time integration of viscous terms. For an adaptive solver, explicit treatment of the viscous term is undesirable as areas with high resolution may impose the diffusive time step constraint $\Delta t \propto \Delta x^2$, where Δx is the non-dimensional spacing. Moreover, excluding several less-favored SPH projection methods [17, 34, 36–38] - which we point out all used uniform meshfree grids - most SPH implementations supplement the continuity equation with an artificial, stiff equation of state in order to avoid the pressure Poisson problem. These approaches are commonly referred to as weakly compressible SPH (WCSPH). Weakly compressible fluid models introduce an restrictive time step proportional to the Mach number, $Ma = U_{max}/c_s$ with c_s the artificial speed of sound. While circumventing the pressure Poisson problem, the relaxation of the continuity constraint introduces Mach number dependence to transient simulations and requires stringent conditions ($Ma < 0.1$). In fact, without grid regularization, Lee et al. reports WCSPH requires $Ma \approx 0.01$ to avoid development of severe voids in the domain [34].

On top of these issues, simulating fluid flow from a strictly Lagrangian viewpoint, as is done in SPH, poses technical challenges due to the accompanying grid distortion. Depending on the flow conditions, grid points following pathlines will lead to undesirable clustering and or voids. See for instance the examples of degenerate Lagrangian grids in Oger et al. [19]. Particle overlap, voids, and high grid irregularity all lead to instabilities for consistent meshfree methods which must rely on a local matrix inversion to construct approximations. For mesh-based methods, these same issues arise and led to the development of arbitrary Lagrangian-Eulerian (ALE) methods [39]. More recently, the grid degradation problem has been re-addressed in SPH [7, 19, 37, 38, 40].

Specifically, Chaniotis et al. proposed periodically rezoning the Lagrangian particles

and remapping their properties onto a uniform grid [40]. In their incompressible SPH solver, Xu et al. proposed to rezone Lagrangian particles with a regularizing shift and remap their properties [37]. Hu et al. modified this regularizing shift to ensure adaptivity in their weakly compressible solver [7]. On the other hand, while similarly using a weakly compressible solver, Oger et al. proposed a direct ALE scheme wherein the grid velocity is comprised of the Lagrangian component with an added regularizing velocity [19]. Common to all of these approaches is the limitation that the grid motion is still tied to the fluid flow leading to Reynolds number dependent grid motion.

On a final note, with the notable exception of several adaptive SPH implementations [7, 35, 41–45], nearly all SPH solvers use uniform meshfree grids. Consequently, small spatial scales are difficult if not impossible to adequately resolve. Upon closer examination of the adaptive solvers, we note that all of these solvers use the weakly compressible model and treat the viscous term explicitly (if present). Moreover, excluding the recent work of Hu et al. [7], all approaches partitioned the computational domain into a few predefined refinement zones with the grid spacing changing discontinuously near the interface of adjacent refinement zones. These sharp transitions not only compromise the accuracy but also limit the maximum achievable grid spacing ratio.

1.1.2 Element Free Galerkin Method

Building on Nayroles' et al. Diffuse Element Method (DEM) [46], in 1994, Belytschko et al. introduced the Element-Free Galerkin (EFG) method [1]. In EFG, the weak form of the PDE is discretized using trial and test functions both constructed from the polynomial reproducing Moving Least Squares (MLS) approximation [47]. MLS produces a continuous approximation for the derivatives across the entire domain domain making it a particularly attractive alternative to the traditional finite element basis. However, it is not an interpolation, as such, imposing essential boundary conditions requires added effort (e.g., Lagrange multipliers [48]).

Moreover, as stated by Belytschko et al., "the major dilemma . . . revolves around how to evaluate the integrals . . ." [2]. That is, the evaluation of the discretized weak form integrals requires re-introduction of a suitable background mesh with appropriate integration points [31, 49]. Unlike in the Finite Element (FE) method [50], where local element matrices are assembled and summed into the global matrix, in EFG, local matrices are computed for each integration point before being summed into the global system. This is due to the fact that each integration point may have a different connectivity.

Consequently, this method, while certainly promising due to the fact the trial and test functions are not tied to any mesh, is still cumbersomely tied to the quadrature points of a background mesh. Moreover, Beissel et al. point out that when a simple nodal integration scheme is used, spurious modes will arise unless a stabilization procedure is applied [51].

EFG appears to mostly have been applied in the solid mechanics disciplines; however, a decade after the introduction of the EFG method, Huerta et al. [52] extended EFG method to incompressible flows in their pseudo-divergence free (PDF) EFG method. Their approach allows for equal order interpolation of velocity and pressure as is evident by the lack of spurious pressure modes in their 2D steady Stokes flow test case.

As an aside, note that the underlying meshfree approximation in EFG, that is MLS, can also be tied to the quadrature points of a finite volume (FV) scheme on some background mesh. In 2007, Cueto-Felgueroso et al. proposed an approach for the compressible Navier-Stokes equations based on MLS [53]. In their approach, the nodes used to construct the MLS approximation coincide with the nodes of the cell-centered FV scheme and so the MLS approximation is directly coupled to the mesh; however, this need not be the case, though it is a reasonable choice. More recently, the approach was extended to incompressible flow on fixed, unstructured grids [54].

1.1.3 Reproducing Kernel Particle Method

In 1995, Liu et al. proposed a continuous and corresponding discrete correction function to SPH kernels in their Reproducing Kernel Particle Method (RKPM) [5]. Under certain conditions, the consistent approximation in RKPM coincides with the MLS approximation and so will result in equivalence between RKPM and EFG [2, 5]. Indeed, their original paper also uses RKPM with a Galerkin formulation wherein boundary conditions are implemented using Lagrange multipliers. Concurrently, they developed multiscale methods based on the reproducing kernel and wavelets [55]. In subsequent work, Chen et al. applied RKPM to large deformation analysis of non-linear structures [56], whereas Aluru studied RKPM with a collocation technique [57]. An overview of RKPM and its applications is given by Liu et al. [58].

1.1.4 General Finite Differences

In 1996, Onate et al. introduced the Finite Point Method (FPM) and applied it to 2D compressible flow over a NACA airfoil using a clustered, fixed meshfree grid [12, 59]. In FPM, instead of discretizing the weak form, the strong form of the PDE is discretized using the collocation method with the polynomial reproducing weighted least squares (WLS) approximation [60]. To stabilize the convective transport, a residual-based stabilization technique was applied.

Applying a WLS discretization to the strong form of a PDE is not new. In fact, Onate et al.'s work was preceded by Batina, who used a least squares (LS) fit to compute 2D and 3D compressible flow solutions on meshfree grids constructed from unstructured triangular meshes [61]. Dating even further back, building on Perrone and Kao's work [62], Liszka and Orkisz applied WLS to solve various problems in mechanics nearly two decades earlier [10]. As others have before us, in this work, we will refer to the approach of applying a point collocation with WLS or LS, as General Finite Differences (GFD). The reason for this is made more clear in Section 2.1.

Over the years a number of publications have proposed using LS or WLS with a strong form discretization (i.e., GFD) [24, 63–70]. Aluru and Li proposed to use a fixed kernel approximation in their Finite Cloud Method (FCM) [63]. As they point out, when the nodal volume is set to unity the fixed kernel approximation coincides with WLS. They applied the approach to several 1D and 2D problems in heat conduction and thermoelasticty.

With respect to viscous, incompressible flows, Ding et al. applied a LS discretization to a stream function-vorticity formulation on fixed, hybrid meshfree-Cartesian grids [66]. They simulated 2D flow past a cylinder for Reynolds number up to $Re = 200$. Mendez and Velazquez applied a LS discretization with a Lax-Wendroff time integration scheme to the artificial compressibility method [67]. They used the approach with fixed meshfree grids to simulate 2D unsteady flow over stationary bodies. On the other hand, in Tiwari and Kuhnert's Finite Pointset Method (FPM), the WLS approximation is used to discretize an explicit first order approximate projection method. They simulate several standard benchmarks (i.e., Couette flow, Poiseuille flow, lid driven cavity) while using Lagrangian meshfree grids [68].

As mentioned earlier, integrating along the Lagrangian path leads to grid degeneration. In fact, Jin et al. have pointed out collocated meshfree methods are particularly sensitive to distribution of the grids points [69]. To remove this sensitivity and improve robustness, Jin et al. have proposed a set of techniques to enforce the positivity conditions. These conditions are closely related to the convergence behavior of finite difference solutions for elliptic problems as well as to whether or not the underlying numerical solution is physical [71, 72].

Others, such as Chew et al. [24], instead circumvented the grid degeneration problem by using WLS with an ALE formulation. They used the approach to simulate incompressible flow past moving bodies on hybrid meshfree-Cartesian grids. In their approach, a layer of meshfree ALE nodes follows the solid body and overlaps a background Cartesian grid. Their approach treats viscous terms implicitly, uses a fractional step algorithm to

approximately enforce incompressibility, and has a low memory bandwidth. However, the approach is complicated by freshly cleared Cartesian nodes and the use of two overlapping grids. Additionally, the use of grid adaptivity is limited.

With respect to incompressible 3D flows, Wang et al. applied the WLS discretization to a semi-implicit approximation projection method and simulated flow past a sphere for Reynolds numbers of 100, 250, and 300 using fixed, clustered meshfree grids [70].

1.1.5 h-p clouds

In 1996, Duarte and Oden [73] proposed h-p clouds, an approach based on the partition of unity (PU) framework. Similar to EFG, in the original paper, the family of approximations in h-p cloud are used to discretize a Galerkin formulation. Unlike the pure MLS approximation, in the h-p clouds approximation, the order of the polynomial basis can vary throughout the domain without introducing discontinuities in the approximation [2]. Their results show that using a zeroth order basis for the partition of unity (i.e., the Shepard functions) is most efficient as it will still preserve the underlying polynomial approximation order and does not require a matrix inversion. In addition to flexible hp adaptivity, the PU approach can also be used to supplement the solution space with an extrinsic basis specific to the problem at hand [74].

The key ideas present in h-p clouds were also independently proposed by Babuska and Melenk in their Partition of Unity Finite Element Method (PUFEM) [75]. In fact, shortly after, these approaches seem to have coalesced as the Generalized Finite Element (GFE) method [76].

1.1.6 Meshless Local Petrov-Galerkin

In 1998, Alturi and Zhu introduced the Meshless Local Petrov-Galerkin method [9]. Unlike EFG, in MLPG, the weak form of the PDE is discretized using distinct trial and test functions. The former correspond to MLS shape functions, whereas the latter correspond

to the weight function used in the MLS approximation. To evaluate integrals, instead of introducing a background mesh, integration points from the local quadrature domain (e.g., circle in 2D) are directly used. They verified the approach on the 2D Poisson equation and potential flow around a cylinder. Several years later, Wu et al. [77] applied a modified MLPG approach to simulate 2D natural convection in a concentric annulus using a vorticity-stream function formulation on fixed, clustered meshfree grids using a staggering 64 integration points per quadrature domain. Such a large number of integration points indicates the difficulty of accurately resolving the integrals.

1.2 Thesis objectives

The goal of this thesis is to develop a collocated meshfree solver for viscous incompressible flows involving moving interfaces and adaptive refinement. To this end, several of the existing limitations mentioned in the overview need to be address. These limitations are summarized and addressed as follows:

1. **Incompressibility**. Weakly compressible solvers introduce Mach number dependence into the solution through a numerical speed of sound in the artificial equation of state. To avoid this tuning parameter and corresponding restrictive time step, we will apply the approximate projection method. The approach handles pressure implicitly, removes checkerboarding due to co-located pressure and velocity, and approximately enforces the divergence free condition.

2. **Arbitrary Lagrangian-Eulerian (ALE) formulation.** Pure Lagrangian formulations are not suitable for simulating fluid flow due to the accompanying grid distortion. Three stage schemes comprised of a Lagrangian stage with subsequent rezoning and remapping stages have found success in many SPH solvers; however, these approaches are limited in that the grid motion is still coupled to the fluid flow leading to Reynolds number dependent grid motion. Recognizing three stage schemes are

indirect ALE solvers, we instead consider the ALE formulation of the incompressible Navier-Stokes equations and investigate a grid motion strategy that is decoupled from the fluid flow.

3. **Meshfree grid generation, regularization, and adaptivity**. Quasi-uniform grids waste computational resources and are limited in the spatial resolution they can resolve. As such, grid clustering should be introduced to resolve the rapidly changing flow near relevant boundaries. To remove the restrictive diffusive time step associated with fine resolutions, the viscous terms should be handled implicitly. To generate high quality adaptive meshfree grids, a general approach is needed that handles complex boundaries, enforces a regular grid, and simultaneously ensures the gid spacing smoothly transitions from low resolutions regions to high resolution regions. Persson's simple mesh generator [78] is one such approach. Here, this approach will be applied in a meshfree fashion.

4. **Consistency**. To ensure truncation errors that go to zero, the spatial discretization should enforce the polynomial reproducing conditions. Moreover, to lower computational costs, the approximation should also result in compact stencils with a low number of nearest neighbors. To enforce these conditions, we will apply the weighted least squares approximation.

The thesis overview is as follows.

1.3 Thesis overview

We first introduce the methodology in Chapter 2. Specifically, in Section 2.1, we introduce the collocation method which can be used with any number of meshfree approximations to discretize PDEs. In Section 2.2, we introduce several meshfree approximations that satisfy the polynomial reproducing conditions and so can be used to discretize PDEs in a consistent manner. Subsequently, in Section 2.3 we detail how to properly measure errors

on quasi-uniform and non-uniform meshfree grids. Afterwards, in Section 2.4, we present details regarding how to solve the incompressible Navier-Stokes equations when variables are co-located by using the approximate projection method. Section 2.5 presents details regarding moving grids and the arbitrary Lagrangian-Eulerian formulation of the incompressible Navier-Stokes equations. The methodology is concluded in Section 2.6, with a discussion of how to generate high quality meshfree adaptive grids.

Following the methodology, results are presented in the remaining chapters and are split as follows. In Chapter 3, results on fixed, meshfree grids are presented for uniform and non-uniform cases. In Section 3.1, we study the Poisson equation with several verification benchmarks presented for both simple and complex geometries in 2D with deferred corrections and one example in 3D with h adaptivity. In Section 3.3, validation benchmarks are presented for incompressible flows, namely the classic lid driven cavity and uniform flow over a cylinder. For both of these benchmarks, isotropic clustering is used near the boundary to resolve the viscous flow.

In Chapter 4, results on moving uniform meshfree grids are presented. Both verification and validation benchmarks are presented. Specifically, we verify the moving grid solver on the periodic Taylor-Green vortex decay problem (Section 4.1) and on the modified lid driven cavity problem (Section 4.2). As for validation, in Section 4.3, we study the inline oscillating cylinder. Varying the Reynolds numbers, results are validated against high resolution lattice Boltzmann method simulations as well as to results available in the literature.

In Chapter 5, results on moving adaptive grids are presented. Specifically, we consider two validation benchmarks with isotropic h adaptivity near boundaries. We first revisit the inline oscillating cylinder in Section 5.1 to validate the approach. Subsequently, the domain is extended to mimic an unbounded domain and the flow is validated against experimental and numerical results. The next validation benchmark considered is more challenging. In our final benchmark, we validate the drag and lift on two cylinders as they move towards

and pass each other. Conclusions and future outlook are discussed in Chapter 6.

CHAPTER 2

METHODOLOGY

2.1 Collocation method

Consider the Poisson equation

$$\begin{aligned} \mathcal{L}u\left(\mathbf{x}\right) &= f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \\ \frac{\partial u}{\partial n} &= g(\mathbf{x}), \quad \mathbf{x} \in \Gamma_n, \\ u\left(\mathbf{x}\right) &= h(\mathbf{x}), \quad \mathbf{x} \in \Gamma_d, \end{aligned} \tag{2.1}$$

where $\mathcal{L}$ is a differential operator taken here as the Laplacian $\mathcal{L} = \nabla^2$ and $u\left(\mathbf{x}\right)$ is the unknown scalar field. In the method of weighted residuals, a trial function $u^h\left(\mathbf{x}\right)$ is assumed and the PDE is integrated over the domain with respect to a test (weight) function $v\left(\mathbf{x}\right)$

$$\int_\Omega r^h\left(\mathbf{x}\right) v\left(\mathbf{x}\right) d\Omega = 0, \tag{2.2}$$

where $r^h\left(\mathbf{x}\right)$ is the residual

$$r^h\left(\mathbf{x}\right) = \mathcal{L}u^h\left(\mathbf{x}\right) - f(\mathbf{x}), \tag{2.3}$$

with the trial function defined as

$$u^h\left(\mathbf{x}\right) = \sum_i^N \phi_i\left(\mathbf{x}\right) \hat{u}_i, \tag{2.4}$$

where $\phi_i\left(\mathbf{x}\right)$ are the global basis and $\hat{u}_i$ are the unknown nodal coefficients that need to be determined. Differences regarding what global basis is chosen and what test function space is chosen leads to different numerical methods. In the collocation method, integrals

are entirely avoided. To determine the nodal coefficients, the residual is set to be orthogonal to the shifted Dirac delta distribution $\delta(\mathbf{x} - \mathbf{x}_j)$. For the above Poisson problem, this gives the following linear system of equations

$$\begin{aligned}
\sum_j^N \nabla^2 \phi_j\big|_{\mathbf{x}_i} \hat{u}_j &= f(\mathbf{x}_i), \quad \mathbf{x}_i \in \Omega, \\
\sum_j^N \left.\frac{\partial \phi_j}{\partial n}\right|_{\mathbf{x}_i} \hat{u}_j &= g(\mathbf{x}_i), \quad \mathbf{x}_i \in \Gamma_n, \\
\sum_j^N \phi_j(\mathbf{x}_i)\, \hat{u}_j &= h(\mathbf{x}_i), \quad \mathbf{x}_i \in \Gamma_d,
\end{aligned} \tag{2.5}$$

where note that we have swapped j and i such that i denotes a row in the the final matrix and j a column. For a given row i, the stencil coefficients are simply the global basis differentiated appropriately and evaluated at $\mathbf{x}_i$. Although certainly not a requirement, the global basis can be chosen to result in a compact stencil, that is $\phi_j(\mathbf{x})$ is only non-zero for $\mathbf{x}$ within some neighborhood around $\mathbf{x}_j$. This changes the global computation over all j into a local one over only the neighbors of point $\mathbf{x}_i$ that contribute a non-zero value. Moreover, the global basis $\phi_i(\mathbf{x})$ is often chosen to satisfy the Kronecker delta property (i.e., form an interpolation)

$$\phi_i(\mathbf{x}_j) = \delta_{ij}, \tag{2.6}$$

with $\delta_{ij} = 1$ when $i = j$ and zero otherwise. Consequently, the approximation at $\mathbf{x}_i$ coincides with the nodal coefficients (i.e., $u^h(\mathbf{x}_i) = \hat{u}_i$).

There are several disadvantages of using a collocation method as compared to the Galerkin approach. The first is that the approach retains higher derivatives as no integration by parts can be used. Second, the residual is controlled only at a set of discrete grid points. Lastly, the collocation method generally leads to an asymmetric Laplacian matrix.

Despite these short comings, in the meshfree setting, the approach is cheaper than the Galerkin approach and easier to implement. Specifically, the matrix assembly is a simple

node by node assembly that only requires a local moment matrix to be inverted once per node whereas a meshfree Galerkin approach such as EFG will require assembling local contributions from each quadrature point, which in turn requires inverting a moment matrix at each integration point. On top of this, the collocated approach is entirely decoupled from use of any background mesh, making it particularly flexible for use with moving grids.

So far we have assumed that the trial function is formed using a global basis. However, this is not always the case. More specifically, it is also possible to use a local basis. By local we do not mean in the sense of the computation (a global approximation can be locally computed) but in the sense that the approximation is only accurate in the neighborhood (cloud) of a certain point and in general is multi-valued.

Correspondingly, these local approximations are used with local collocation. That is, a local residual is constructed about a given point's neighborhood (cloud) as opposed to everywhere in the domain

$$\mathbf{r}_i^h(\mathbf{x}) = \mathcal{L}u_i^h(\mathbf{x}) - f(\mathbf{x}), \tag{2.7}$$

where a subscript i has been added to $u_i^h(\mathbf{x})$ to indicate it is a local approximation constructed about point $\mathbf{x}_i$ using n neighboring points

$$u_i^h(\mathbf{x}) = \sum_j^n \phi_j(\mathbf{x})\,\hat{u}_j. \tag{2.8}$$

Here, $\phi_j(\mathbf{x})$ are local to the node i and do not form a global basis. Nor do they typically satisfy the Kronecker delta property (i.e., $u_i^h(\mathbf{x}_i) \neq \hat{u}_i$). This local basis is usually referred to as shape functions. For example, in the finite element context, the shape functions are local to the element and collectively form the global basis (see e.g., Figure 2.1b). However, in our context, these shape functions do not form a global basis and the local approximations are generally discontinuous in the overlapping region of clouds corresponding to different points (see e.g., Figure 2.2 and Figure 2.1 in the following section).

To obtain the nodal coefficients $\hat{u}_i$, for every point $\mathbf{x}_i$ the respective residual is set to

zero $r_i^h(\mathbf{x}_i) = 0$ resulting in one equation per point. The resulting system of equations is equivalent to Eq. 2.5 and can be compactly written as

$$\mathbf{L}^h \hat{\mathbf{u}} = \mathbf{b}, \tag{2.9}$$

where now local shape functions are used as opposed to a global basis. With polynomial shape functions on structured grids, this approach is analagous to the well known finite difference method. Consequently, local point collocation on meshfree grids with polynomial shape functions is typically refered to as General Finite Differences (GFD). As we will see next, the shape functions must satisfy certain conditions in order for the local truncation error to be consistent. In Section 2.2.1, we will show these conditions are automatically satisfied if the shape functions are constructed to satisfy the polynomial reproducing conditions [2].

2.1.1 Local truncation error

The nodal coefficients $\hat{\mathbf{u}}$ will exactly satisfy the discrete system; however, the exact solution to the PDE (Eq. 2.1) will not generally satisfy the discrete equations given in Eq. 2.5. How closely the exact solution satisfies the discrete system is described by the local truncation error τ.

While it is not immediately obvious, the local truncation error τ is itself related to the error between the discrete and exact solutions. In fact, if the discretization is stable, the error will converge at the same rate as the local truncation error [79]. If the local truncation error scales with the grid spacing h through some power k (e.g., $\tau_i = O(h^k)$) then the discretization is said to be consistent to the k^{th} order.

If we restrict the Poisson problem to 1D, the exact solution $u(x)$ will satisfy the fol-

lowing equation for an interior point i

$$\sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} u_j - f_i = \tau_i \tag{2.10}$$

where $u_j = u(x_j)$ is the exact solution to Eq. 3.1 evaluated at the x_j position and τ_i is the local truncation error at x_i. We can perform a Taylor series expansion for u_j about u_i to obtain the local truncation error at point x_i in terms of exact solution and its derivatives. Specifically, we have the following relation for the local truncation error

$$\begin{aligned}\tau_i = u_i \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} + \left.\frac{du}{dx}\right|_{x_i} \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji} + \left.\frac{1}{2!}\frac{d^2u}{dx^2}\right|_{x_i} \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^2 \\ + \left.\frac{1}{3!}\frac{d^3u}{dx^3}\right|_{x_i} \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^3 + \left.\frac{1}{4!}\frac{d^4u}{dx^4}\right|_{x_i} \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^4 \\ + \ldots \left.\frac{1}{k!}\frac{d^ku}{dx^k}\right|_{x_i} \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^k - f_i,\end{aligned} \tag{2.11}$$

where $x_{ji} = x_j - x_i$ and where note the second derivatives of the shape functions are inversely proportional to the squared grid spacing h

$$\frac{d^2\phi_j}{dx^2} \propto \frac{1}{h^2} \tag{2.12}$$

With this clarified, it is clear that if we desire that the local truncation error to be second order accurate for the above example, then the shape functions must satisfy the following discrete conditions

$$\sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} = 0 \quad \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji} = 0 \quad \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^2 = 2 \quad \sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^3 = 0, \tag{2.13}$$

where the third condition $\sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} x_{ji}^2 = 2$ is needed to get $\left.\frac{d^2u}{dx^2}\right|_{x_i} - f_i = 0$. If the shape

functions satisfy these conditions then the local truncation error is

$$\tau_i = \frac{1}{4!}\frac{d^4u}{dx^4}\bigg|_{x_i} \sum_j^n \frac{d^2\phi_j}{dx^2}\bigg|_{x_i} x_{ji}^4 + \ldots \frac{1}{k!}\frac{d^ku}{dx^k}\bigg|_{x_i} \sum_j^n \frac{d^2\phi_j}{dx^2}\bigg|_{x_i} x_{ji}^k. \tag{2.14}$$

It is well known that the centered difference approximation satisfies the conditions stated in Eq. 2.13 on a uniform grid. This results in a local truncation error that is second order

$$\tau_i \approx \frac{1}{12}h^2\frac{d^4u}{dx^4}\bigg|_{x_i}. \tag{2.15}$$

For finite differences, the various approximations are usually derived from the method of undetermined coefficients which usually obscures the connection to the polynomial reproducing conditions. Indeed, the approximation power of centered differences directly comes from the fact that the stencil coefficients correspond to appropriately differentiating local quadratic shape functions that reproduce polynomials up to degree two. These shape functions are shown in Figure 2.1.

In Section 2.2.1, we show that if the shape functions reproduce up to degree two polynomials then the first three conditions in Eq. 2.13 are automatically satisfied. The last condition in Eq. 2.13 is not enforced for degree two reproduction. However, since the 1D quadratic shape functions shown in Figure 2.1 not only satisfy the positivity condition $\frac{d^2\phi_j}{dx^2}\Big|_{x_i} > 0$ for the two neighbors but also result in equal coefficients, then using fact the local truncation error is an odd moment (i.e., the x_{ji} is raised to an odd power), the error term will cancel on uniform grids and the FD approximation will gain an order. For non-uniform grids this cancellation will not generally be the case. As such, for this 1D example, the truncation error will generally be first order for an interior point on irregular grids

$$\tau_i \approx \frac{1}{3!}\frac{d^3u}{dx^3}\bigg|_{x_i} \sum_j^n \frac{d^2\phi_j}{dx^2}\bigg|_{x_i} x_{ji}^3 \approx O(h). \tag{2.16}$$

In a similar fashion, the local truncation error can be determined for a point at the

boundary. For example, for a right boundary (i.e., $i = R$) with a Neumann condition imposed, the local truncation error is

$$\begin{aligned}\tau_i = u_i \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} + \left.\frac{du}{dx}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji} + \left.\frac{1}{2!}\frac{d^2u}{dx^2}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^2 \\ + \left.\frac{1}{3!}\frac{d^3u}{dx^3}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^3 + \left.\frac{1}{4!}\frac{d^4u}{dx^4}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^4 \\ + \cdots \left.\frac{1}{k!}\frac{d^ku}{dx^k}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^k - g_i,\end{aligned} \tag{2.17}$$

where note that

$$\frac{d\phi_j}{dx} \propto \frac{1}{h} \tag{2.18}$$

and so the conditions to have a second order local truncation error are

$$\sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} = 0 \quad \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji} = 1 \quad \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^2 = 0 \tag{2.19}$$

where the condition $\sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji} = 1$ is needed to satisfy $\left.\frac{du}{dx}\right|_{x_i} - g_i = 0$. If the shape functions satisfy the above discrete conditions then the local truncation error will generally be

$$\tau_i \approx \left.\frac{1}{3!}\frac{d^3u}{dx^3}\right|_{x_i} \sum_j^n \left.\frac{d\phi_j}{dx}\right|_{x_i} x_{ji}^3. \tag{2.20}$$

The conditions imposed in Eq. 2.19 will also be automatically satisfied if the shape functions satisfy the polynomial reproducing conditions up to degree two.

For Dirichlet conditions, a similar analysis can be done if local approximations are also constructed at these known boundary points. In this case, the conditions are just the polynomial reproducing conditions. However, note that it is possible to simply choose a different local approximation that satisfies the Kronecker delta property and avoid errors at Dirichlet boundaries entirely. Specifically, this amounts to setting the diagonal entry to one in the linear system and zero otherwise for rows corresponding to Dirichlet boundary

conditions.

On a final note, while we have restricted our analysis to 1D, it is straightforward to carry out the same procedure for higher dimensions using the multi-dimensional Taylor series, see e.g., Section 3.1.1. Moreover, the same connection between the polynomial reproducing conditions and local truncation error can be made. Before providing details as to how to compute the shape functions on meshfree grids that enforce polynomial reproducing conditions and what local truncation error to expect, we first briefly discuss the connection between the local truncation error, the pointwise error, and the deferred correction approach as is discussed by LeVeque for standard finite differences [79]. As we will show with numerical examples, this same approach carries over to General Finite Differences.

2.1.2 Pointwise error

Sticking with the 1D Poisson problem, if we subtract $\sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} \hat{u}_j - f_i = 0$ from Eq. 2.10, we can obtain the following relation for the pointwise error $e_i = u_i - \hat{u}_i$ for an interior point

$$\sum_j^n \left.\frac{d^2\phi_j}{dx^2}\right|_{x_i} e_j = \tau_i, \tag{2.21}$$

where similar equations can be obtained for the boundary. In fact, we see the same operator $\mathbf{L}^h$ that acts on the discrete solution also acts on the error just with a different right hand side

$$\mathbf{L}^h \mathbf{e}^h = \tau^h, \tag{2.22}$$

where a superscript h has been added to indicate its with regard to a specific grid spacing. If the inverse $(\mathbf{L}^h)^{-1}$ exists and is stable as h goes to zero then the error will converge to the exact solution at least at the same rate as the truncation error [79]. Stability here refers to the requirement that for all grid spacings small enough, $||(\mathbf{L}^h)^{-1}||$ is bounded by a constant C that is independent of the grid spacing h.

For finite differences on irregular grids, Demkowicz et al. have shown that this is indeed

the case as long as the stencil coefficients satisfy certain conditions [71]. They report that a local quadratic approximation will give the following theoretical error bound for point collocation on elliptic problems

$$\max_i |u(\mathbf{x}_i) - u^h(\mathbf{x}_i)| \leq Ch, \tag{2.23}$$

where $\mathbf{x}_i$ denotes a particular star node. They confirm their results numerically, indicating that in practice the measured convergence rate is one order higher (i.e., $O(h^2)$). We revisit this peculiarity with a numerical example in Section 3.1.1.

2.1.3 Deferred correction

Deferred corrections [79] are a useful technique to increase the accuracy of the approximation. The idea is that if we know the truncation error τ^h, then we can solve Eq. 2.22 to compute the exact error at the given resolution $\mathbf{e}^h = \left(\mathbf{L}^h\right)^{-1} \tau^h$. This exact error can then be added back to the discrete solution to obtain the exact solution. If the truncation error is known this can be simplified in one step

$$\mathbf{L}^h \mathbf{u} = \mathbf{b} + \tau^h. \tag{2.24}$$

However, the truncation error is generally not known as it depends on various derivatives of the solution itself. Instead, we may construct an estimate $\hat{\tau}^h \approx \tau^h$ and solve for the approximate error

$$\hat{\mathbf{e}}^h = \left(\mathbf{L}^h\right)^{-1} \hat{\tau}^h, \tag{2.25}$$

which can be used to correct the discrete solution

$$\mathbf{u}^* = \hat{\mathbf{u}} + \hat{\mathbf{e}}^h. \tag{2.26}$$

To construct the estimate $\hat{\tau}^h$, it is necessary to first determine the form of local truncation error term for interior points and boundary points. The form of the local truncation error will depend on the operators in the PDE and on the polynomial reproducing conditions chosen to be enforced. Once we know the form of the local truncation error, we can construct an estimate by first solving the PDE to obtain an uncorrected solution. This low order solution can then be used to approximate higher order derivatives in the truncation error. The lower order solution can then be corrected using the approximate error given by Eq 2.25. This step requires a second solve but will lead to a more accurate solution. In Section 3.1.1 and Section 3.1.2, examples are shown of how this approach can be used to obtain higher order approximations while still using cheap lower order stencils.

2.2 Meshfree approximation

In this section, we provide some details regarding some common meshfree approximations: least squares (LS), weighted least squares (WLS), moving least squares (MLS), and partition of unity (PU). In principle, any of these scattered data approximations [80] can be used with a meshfree collocation method as they all can enforce the polynomial reproducing conditions. The approach that we have chosen to use, mostly due to its simplicity, is WLS. When WLS is combined with the local point collocation method described in the previous section, the approach is referred to as the General Finite Difference (GFD) method. Before discussing these popular approximations, we first introduce the polynomial reproducing conditions which are central to the consistency of any of these approximations.

2.2.1 Polynomial reproducing conditions

Sticking with 1D, if we require the approximation $u^h(x) = \sum_j \phi_j(x)\hat{u}_j$ to reproduce polynomials of degree k, then the following $k+1$ conditions must be enforced by the shape

functions

$$1 = \sum_j \phi_j(x) \quad x = \sum_j \phi_j(x)\, x_j \quad x^2 = \sum_j \phi_j(x) x_j^2 \quad \ldots \quad x^k = \sum_j \phi_j(x) x_j^k \tag{2.27}$$

For simplicity, if we assume the shape functions reproduce $k = 2$ polynomials (P2), then after differentiating the polynomial reproducing conditions, we obtain the additional conditions that are automatically satisfied

$$\begin{aligned} 0 &= \sum_j \frac{d\phi_j(x)}{dx} \quad 1 = \sum_j \frac{d\phi_j(x)}{dx} x_j \quad 2x = \sum_j \frac{d\phi_j(x)}{dx} x_j^2 \\ 0 &= \sum_j \frac{d^2\phi_j(x)}{dx^2} \quad 0 = \sum_j \frac{d^2\phi_j(x)}{dx^2} x_j \quad 2 = \sum_j \frac{d^2\phi_j(x)}{dx^2} x_j^2. \end{aligned} \tag{2.28}$$

These conditions are exactly the same conditions that we saw arising earlier when analyzing the local truncation error at a specific point x_i using a Taylor series. Here, however, they are presented for a polynomial centered about the origin and evaluated at a generic point x. If we shift the monomials about x_i and evaluate the conditions at $x = x_i$, then we obtain the more familiar form we saw earlier

$$\begin{aligned} 1 &= \sum_j \phi_j(x_i) \quad 0 = \sum_j \phi_j(x_i)\, x_{ji} \quad 0 = \sum_j \phi_j(x_i)\, x_{ji}^2 \\ 0 &= \sum_j \frac{d\phi_j(x_i)}{dx} \quad 1 = \sum_j \frac{d\phi_j(x_i)}{dx} x_{ji} \quad 0 = \sum_j \frac{d\phi_j(x_i)}{dx} x_{ji}^2 . \\ 0 &= \sum_j \frac{d^2\phi_j(x_i)}{dx^2} \quad 0 = \sum_j \frac{d^2\phi_j(x_i)}{dx^2} x_{ji} \quad 2 = \sum_j \frac{d^2\phi_j(x_i)}{dx^2} x_{ji}^2 \end{aligned} \tag{2.29}$$

For non-uniform grids, the associated local truncation error for estimates of a function and its first and second derivatives are proportional to the higher order moments that were not

enforced by the approximation

$$\begin{aligned}
\tau_i^0 &\approx \frac{1}{3!}\frac{d^3u}{dx^3}\bigg|_{x_i} \sum_j^n \phi_j\left(x_i\right) x_{ji}^3 \\
\tau_i^1 &\approx \frac{1}{3!}\frac{d^3u}{dx^3}\bigg|_{x_i} \sum_j^n \frac{d\phi_j}{dx}\bigg|_{x_i} x_{ji}^3 \\
\tau_i^2 &\approx \frac{1}{3!}\frac{d^3u}{dx^3}\bigg|_{x_i} \sum_j^n \frac{d^2\phi_j}{dx^2}\bigg|_{x_i} x_{ji}^3
\end{aligned} \tag{2.30}$$

where the superscript on τ_i has been introduced to indicate the derivative with which it is associated. As such, we see there is a direct tie between the shape functions enforcing the polynomial reproducing conditions given by Eq. 2.27 and the degree of consistency of the local truncation error. More generally, if we introduce the vector of shape functions

$$\mathbf{\Phi}^T\left(x\right) = \begin{bmatrix} \phi_1\left(x\right) & \phi_2\left(x\right) & \dots & \phi_n\left(x\right) \end{bmatrix}, \tag{2.31}$$

and use $\mathbf{A}$ defined in the subsequent section (see Eq. 2.36), we can compactly write the discrete conditions that enforce a consistent local truncation error as

$$\frac{d^q\mathbf{\Phi}^T}{dx^q}\bigg|_{x_i} \mathbf{A} = \frac{d^q\mathbf{P}^T}{dx^q}\bigg|_{x_i}, \tag{2.32}$$

where $q \in 0 \dots k$ and where $\mathbf{P}$ is the degree k shifted 1D polynomial basis. Assuming a smooth solution, the local truncation error for the qth derivative to leading order is

$$\tau_i^q \approx \frac{1}{(k+1)!}\frac{d^{k+1}u}{dx^{k+1}}\bigg|_{x_i} \sum_j^n \frac{d^q\phi_j}{dx^q}\bigg|_{x_i} x_{ji}^{k+1} \tag{2.33}$$

Similar formulas can be obtained in higher dimensions.

2.2.2 Weighted least squares

The idea in weighted least squares is to find $u^h(\mathbf{x})$ that is a local approximation (fit) to the field data $\hat{u}_j$ about some fixed point $\mathbf{x}_i$, also commonly referred to as the star node. The simplest choice for the fit is a local polynomial approximation

$$u^h(\mathbf{x}) = \mathbf{P}^T\mathbf{a}, \tag{2.34}$$

where $\mathbf{P}(\mathbf{x})$ is the polynomial basis with m terms and $\mathbf{a}$ are the unknown coefficients that need to be determined. Note we have dropped off the local subscript indicating that this is a local approximation only valid about $\mathbf{x}_i$. Given a set of n nearest neighbors, the following system is constructed for the coefficients $\mathbf{a}$ for the fixed point $\mathbf{x}_i$

$$\mathbf{A}\mathbf{a} = \hat{\mathbf{u}}, \tag{2.35}$$

where $\hat{\mathbf{u}}$ is the the local vector of the n evaluations of the function being fitted and $\mathbf{A}$ is $n \times m$ matrix. When $n = m$, $\mathbf{A}$ corresponds to the square Vandermonde matrix, whose columns correspond to each basis evaluated at the node's position. For example, for a 1D quadratic basis, $\mathbf{A}$ is

$$\mathbf{A} = \begin{bmatrix} 1 & x_1 - x_i & (x_1 - x_i)^2 \\ 1 & x_2 - x_i & (x_2 - x_i)^2 \\ \cdots & \cdots & \cdots \\ 1 & x_n - x_i & (x_n - x_i)^2 \end{bmatrix}. \tag{2.36}$$

Note to avoid the ill-condition system, the quadratic basis has been shifted about the point of interest (i.e., $\mathbf{P}^T = \left[1, x - x_i, (x - x_i)^2\right]$). When the polynomial basis is shifted about point $\mathbf{x}_i$, the coefficients $\mathbf{a}$ now correspond to the Taylor series coefficients at the specified fixed point (i.e., $\mathbf{a}^T = [u(x_i), du/dx|_{x_i}, 0.5d^2u/dx^2|_{x_i}]$).

When $n > m$ (i.e., more neighboring points than there are basis terms), the system

is overdetermined. To solve the overdetermined system we apply weighted least squares (WLS) [60]. Specifically, we constrain the residual vector $\mathbf{r} = \hat{\mathbf{u}} - \mathbf{A}\mathbf{a}$, weighted by a diagonal $n \times n$ matrix $\mathbf{W}$, to be orthogonal to the column space of $\mathbf{A}$ and arrive at the following positive definite system

$$\mathbf{A}^T\mathbf{W}\mathbf{A}\mathbf{a} = \mathbf{A}^T\mathbf{W}\hat{\mathbf{u}}, \tag{2.37}$$

with diagonal weight matrix $\mathbf{W}$ defined as

$$\mathbf{W} = \begin{bmatrix} W(\mathbf{x}_1 - \mathbf{x}_i) & \cdots & \cdots \\ \cdots & W(\mathbf{x}_2 - \mathbf{x}_i) & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & W(\mathbf{x}_n - \mathbf{x}_i) \end{bmatrix}. \tag{2.38}$$

Solving this linear system for the Taylor coefficients $\mathbf{a}$ will give a local approximation that ensures the local residual is minimized with respect to the weighted Euclidean norm.

The weight function is chosen such that the residual corresponding to points near the point of interest (i.e., $\mathbf{x}_i$) are assigned more weight than the other equations. The weight function is restricted to be positive, compact, and symmetric. Note that when $\hat{\mathbf{u}}$ is in the range of $\mathbf{A}$, then the weighting has no effect. If a constant weight is used, then the standard least squares (LS) approximation is obtained. Here, we use the following quintic weight function [11]

$$W(R,h) = \frac{7}{478\pi h^2} \begin{cases} (3-R)^5 - 6(2-R)^5 + 15(1-R)^5 & ,0 \le R < 1, \\ (3-R)^5 - 6(2-R)^5 & ,1 \le R < 2, \\ (3-R)^5 & ,2 \le R < 3, \\ 0 & ,R \ge 3, \end{cases} \tag{2.39}$$

where R is the radius r to a neighboring point normalized by the smoothing length h. Correspondingly, we set the smoothing length such that a specified number of neighbors are found.

Assuming a full column rank $\mathbf{A}$, the final local approximation constructed about point $\mathbf{x}_i$ is formed as the dot product of the shifted polynomial basis $\mathbf{P}$ with the Taylor coefficients $\mathbf{a}$

$$u^h(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W} \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{W} \hat{\mathbf{u}}, \tag{2.40}$$

where the vector of local polynomial shape functions can be defined as

$$\mathbf{\Phi}^T(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W} \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{W}, \tag{2.41}$$

and where the shape function for the jth neighbor (for a given star node i) is

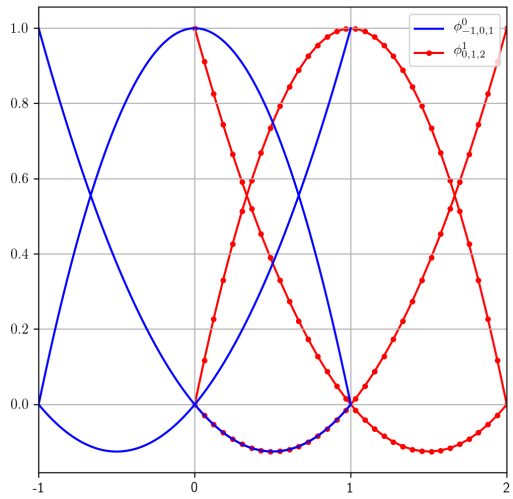
$$\phi_j(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W} \mathbf{A}\right)^{-1} \mathbf{P}(\mathbf{x}_j) W(\mathbf{x}_j - \mathbf{x}_i). \tag{2.42}$$

Note that by construction, the WLS approximation reproduces polynomials up to order k. To verify this note that the polynomial reproducing conditions can be compactly written as

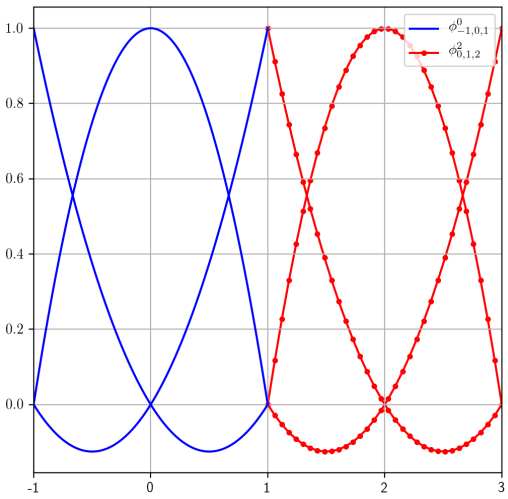
$$\mathbf{P}^T = \mathbf{\Phi}^T(\mathbf{x}) \mathbf{A}. \tag{2.43}$$

Using the definition of the WLS shape functions (Eq. 2.41) in the above equation, we see the equality holds indicating the WLS shape functions reproduces the specified polynomials basis.

Quadratic shape functions are shown in Figure 2.1a for the case of a uniform grid with each node using exactly three neighbors that are weighted uniformly. As can be seen, the shape functions are multivalued [12] and do not form a global basis unless they are constructed in an element-wise manner as done in FE method (see Figure 2.1b). In this latter case, note the global basis is not differentiable everywhere but is integrable.



(a) Quadratic shape functions constructed at nodes 0 (blue) and 1 (red).



(b) Quadratic shape functions constructed on elements [-1,1] (blue) and [1,3] (red).

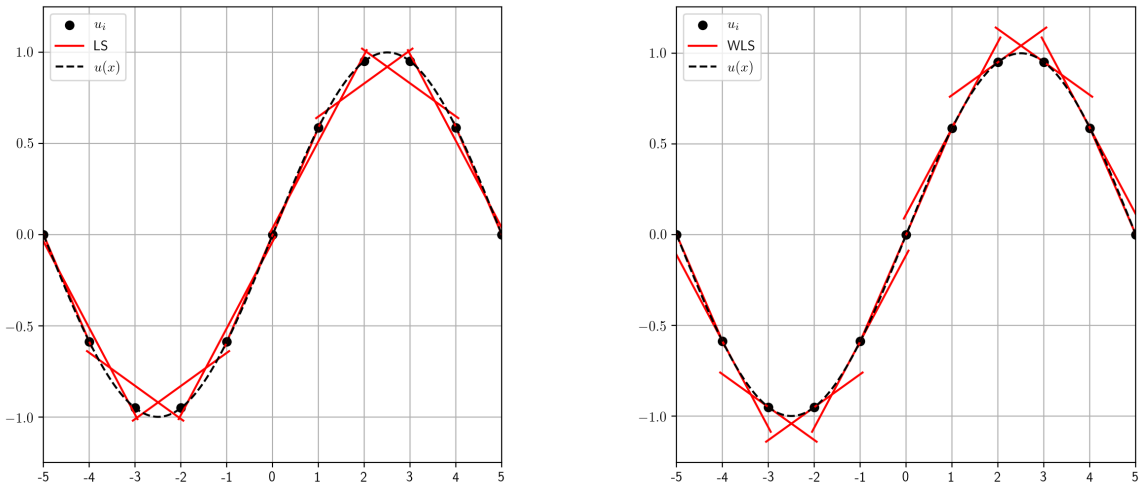
Figure 2.1: Comparison of quadratic shape functions for point clouds constructed about node 0 and node 1 using exactly three neighbors (a) to quadratic shape functions constructed over the elements (b).

While in this simple 1D example, the quadratic shape functions are interpolatory in nature, the WLS shape functions do not generally satisfy the Kronecker delta property (i.e., $u^h(\mathbf{x}_i) \neq \hat{u}_i$). Since the WLS shape functions are polynomials, the approximation $u^h(\mathbf{x})$ is k times differentiable, where k is the order of the polynomial. Unlike MLS, which will be covered next, the weight function differentiability does not influence the approximation's differentiability.

The multivalued nature of the LS or WLS approximation is shown more clearly in Figure 2.2 using a linear approximation. It's evident that the approximation to the function is not global but is instead comprised of overlapping local linear approximations. Each local approximation is constructed about the star node corresponding to u_i. The local approximations are only accurate over their corresponding patches and are most accurate near the point about which they were constructed. Comparing LS and WLS, we see that in WLS, weighting the residual more for points closer to the star node results in the local fit being pulled towards the corresponding data value.

As shown later, using a partition of unity a global basis could be formed that would

preserve the local character of the fits; however, this is not done for WLS as it is typically used with local point collocation wherein the multivalued approximation stemming from neighboring points are simply ignored [12].



(a) LS linear fit.

(b) WLS linear fit.

Figure 2.2: Least squares (a) and weighted least squares (b) linear fit of a sampled sinusoid on a uniform grid using a cloud radius that encloses 3 neighboring points.

In this work, the highest order PDE terms considered are second order and so quadratic WLS shape functions are used which leads to 6×6 local moment matrices in 2D and 10×10 moment matrices in 3D. For a quadratic polynomial basis in 2D, the first derivative $\partial/\partial y$ evaluated at $\mathbf{x}_i$ can be computed as

$$\left.\frac{\partial u^h}{\partial y}\right|_{\mathbf{x}_i} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1} \mathbf{A}^T\mathbf{W}\hat{\mathbf{u}}, \tag{2.44}$$

whereas the second derivative $\partial^2/\partial x^2$ evaluated at $\mathbf{x}_i$ can be computed as

$$\left.\frac{\partial^2 u^h}{\partial x^2}\right|_{\mathbf{x}_i} = \begin{bmatrix} 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix} \left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1} \mathbf{A}^T\mathbf{W}\hat{\mathbf{u}}. \tag{2.45}$$

where note that $\mathbf{A}$ corresponding to a quadratic basis in 2D is

$$\mathbf{A} = \begin{bmatrix} 1 & x_1 - x_i & (y_1 - y_i) & (x_1 - x_i)^2 & (x_1 - x_i)(y_1 - y_i) & (y_1 - y_i)^2 \\ 1 & x_2 - x_i & (y_2 - y_i) & (x_2 - x_i)^2 & (x_2 - x_i)(y_2 - y_i) & (y_2 - y_i)^2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_n - x_i & (y_n - y_i) & (x_n - x_i)^2 & (x_n - x_i)(y_n - y_i) & (y_n - y_i)^2 \end{bmatrix}. \quad (2.46)$$

Observe that the stencil coefficients are simply the corresponding row of $\left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{W}$ scaled by the appropriate Taylor series constant. To obtain the stencil coefficients, we need to compute $\left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1}$ for each star node. For the small matrix sizes, the direct approach in the C++ SMatrix header-only library is particularly efficient [81]. Alternatively, if the system is ill-conditioned a QR factorization can be used [82].

If we remove the constant basis, the size of the matrix $\mathbf{A}^T\mathbf{W}\mathbf{A}$ can be reduced by one row and one column, giving the following local approximation that fits through $\hat{u}\left(\mathbf{x}_i\right)$

$$u^h\left(\mathbf{x}\right) = \hat{u}\left(\mathbf{x}_i\right) + \mathbf{P}^T\left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{W}\left(\hat{\mathbf{u}} - \hat{\mathbf{u}}_i\right), \quad (2.47)$$

where note similar expressions for the stencil coefficients can be obtained. The advantaged of this approach is that the approximation will fit through the value at the star node $\mathbf{x}_i$ and is cheaper to compute (only a 5×5 matrix inverse for a local quadratic approximation in 2D and 9×9 in 3D).

2.2.3 Moving least squares

If the fixed point $\mathbf{x}_i$ is allowed to move (i.e., $\mathbf{x}_i = \mathbf{x}$), then the MLS approximation [1, 12, 47] can be obtained after solving the least squares problem for the the moving coefficients $\mathbf{a}\left(\mathbf{x}\right)$

$$\mathbf{A}^T\mathbf{W}\left(\mathbf{x}\right)\mathbf{A}\mathbf{a}\left(\mathbf{x}\right) = \mathbf{A}^T\mathbf{W}\left(\mathbf{x}\right)\hat{\mathbf{u}}. \quad (2.48)$$

Here, $\mathbf{A}$ is now has a size of $N \times m$, $\hat{\mathbf{u}}$ is no longer the local field data but the global field with size $N \times 1$ and $\mathbf{W}(\mathbf{x})$ is the diagonal weight matrix now a function of $\mathbf{x}$ with size $N \times N$

$$\mathbf{W} = \begin{bmatrix} W(\mathbf{x}-\mathbf{x}_1) & \cdots & \cdots \\ \cdots & W(\mathbf{x}-\mathbf{x}_2) & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & W(\mathbf{x}-\mathbf{x}_N) \end{bmatrix}. \tag{2.49}$$

Due to the compact support of the weight function, the above calculation is still local. However, now the function approximation is global

$$u^h(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W}(\mathbf{x}) \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{W}(\mathbf{x}) \mathbf{u}. \tag{2.50}$$

That is, there is one definition of the trial function that is valid over the entire domain with corresponding global basis

$$\mathbf{\Phi}(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W}(\mathbf{x}) \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{W}(\mathbf{x}), \tag{2.51}$$

where the MLS basis over the jth node is

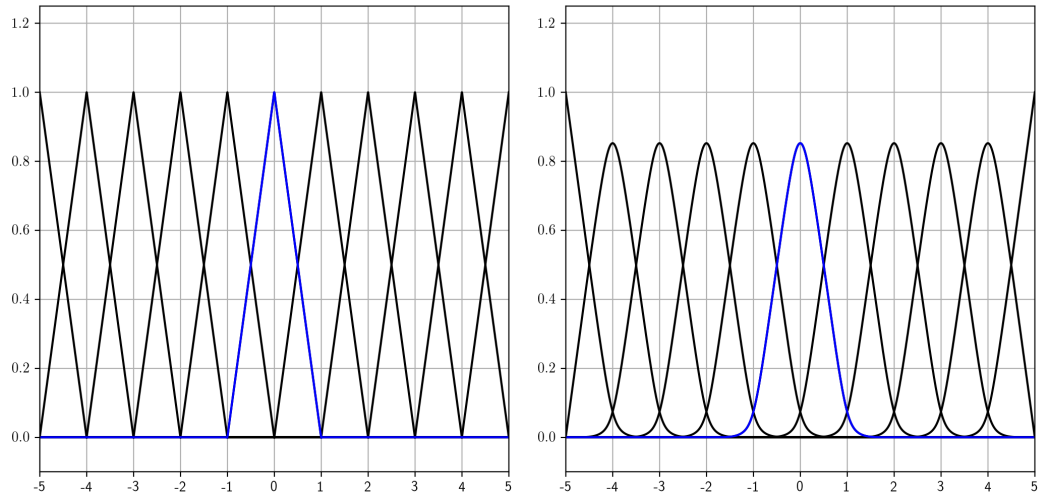
$$\phi_j(\mathbf{x}) = \mathbf{P}^T \left(\mathbf{A}^T \mathbf{W}(\mathbf{x}) \mathbf{A}\right)^{-1} \mathbf{P}(\mathbf{x}_j) W(\mathbf{x}-\mathbf{x}_j). \tag{2.52}$$

In contrast to WLS, to obtain the approximation at $\mathbf{x}$, the moment matrix $\left(\mathbf{A}^T \mathbf{W}(\mathbf{x}) \mathbf{A}\right)^{-1}$ must be inverted for each new position $\mathbf{x}$. For a low order polynomial basis in 1D, 2D, or 3D, explicit formulas can be computed to simplify the calculation [80]; however, for higher order polynomial basis the local moment matrix increases in size (e.g., quadratic basis results in a 6×6 moment matrix in 2D) and can instead be inverted using direct solvers. When taking derivatives, note that $\mathbf{W}(\mathbf{x})$ is a function of $\mathbf{x}$ and so additional terms must be computed [1].

Unlike the WLS approximation, the MLS approximation is not multivalued and forms a global basis as shown in Figure 2.3. For a linear polynomial basis and small cloud radius, the MLS basis resembles the hat function from finite element; however, they are not exactly identical. The MLS basis functions are no longer polynomials. This can be seen more clearly in Figure 2.3b, where a larger smoothing length is used. However, they still - similarly to WLS - enforce the polynomial reproducing conditions. They are k times differentiable, where now k takes the maximum of either the polynomial order or the weight function's continuity.

In the extreme case, a constant polynomial basis can be used and the MLS approximation $u^h(\mathbf{x})$ will still be differentiable as long as the weight function is differentiable. In this case, the shape functions are the well known Shepard functions

$$\phi_j(\mathbf{x}) = \frac{W(\mathbf{x} - \mathbf{x}_j)}{\sum_{i=1}^{N} W(\mathbf{x} - \mathbf{x}_j)}. \tag{2.53}$$



(a) $r = 1.05\Delta x$ (b) $r = 1.9\Delta x$

Figure 2.3: MLS global basis using a cloud radius (a) $r = 1.05\Delta x$ and (b) $r = 1.9\Delta x$ with linear polynomial basis used in $\mathbf{A}$. The blue curve is the MLS basis for node $x = 0$.

In Figure 2.4, the global MLS approximation is compared to the local WLS approxima-

tions for the uniformly sampled sinusoid. To the naked eye, the MLS approximation bears striking resemblance to a piece-wise linear FE interpolation. This is to be expected as the MLS basis functions appeared very similar to the linear FE basis for the chosen parameters. At the points $\mathbf{x}_i$, the local WLS approximation and MLS approximation coincide with large differences in the surrounding neighborhood of point $\mathbf{x}_i$.

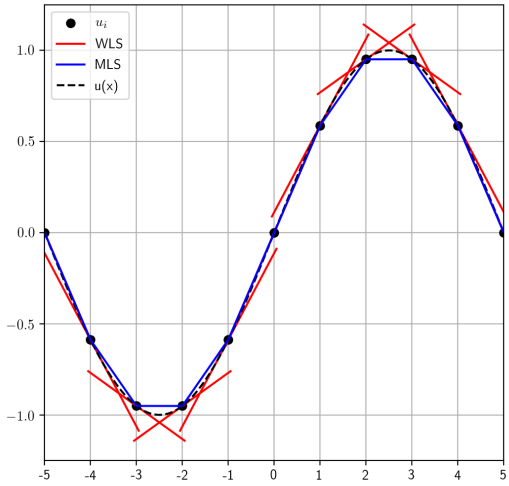


Figure 2.4: MLS v.s. WLS fit of a sampled sinusoid on a uniform grid. Constant cloud radius $r = 1.05\Delta x$ is used, corresponding to 3 neighbors on the interior and 2 neighbors on the boundary, including the central node.

2.2.4 Partition of Unity

The idea in Partition of Unity method (PU) [2, 73, 80], is to partition a domain Ω into N overlapping domains Ω_i that together completely cover the domain Ω. On each domain Ω_i, a continuous function that has compact support is defined such that the sum over all N subdomains sum to one

$$\sum_{i}^{N} \phi_i \left(\mathbf{x}\right) = 1. \tag{2.54}$$

This property is in fact just the constant reproducing condition. The Shepard functions form a partition of unity (PU), as do any of the higher order MLS shape functions [2]. Using the PU method, we can blend together the local WLS fits into a global approximation

while not only preserving the local character of the fit but also the approximation order [80]. Specifically, if we replace the data u_i with the local WLS fit $u_i^h(\mathbf{x})$ corresponding to a given node $\mathbf{x}_i$ then we obtain a global PU approximation

$$u^h(\mathbf{x}) = \sum_{i=1}^{N} \phi_i(\mathbf{x})\, u_i^h(\mathbf{x}). \tag{2.55}$$

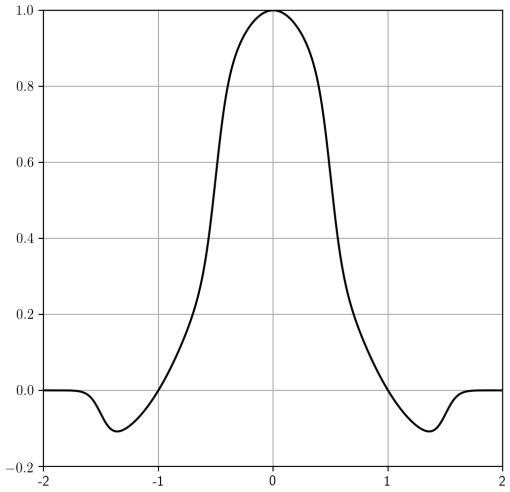


Figure 2.5: PU basis for the quadratic shape functions shown in Figure 2.1a.

In Figure 2.5, the PU basis for node 0 is shown when the local quadratic shapes functions are used. In Figure 2.6, we illustrate the PU approximation when applied to quadratic local WLS approximations that arise from solving the overdetermined case. The PU functions are chosen to be the Shepard functions. Note that the functions $u_i^h(\mathbf{x})$ are not restricted to be local WLS fits as is shown in this example nor are they restricted to use the same basis for each cloud.

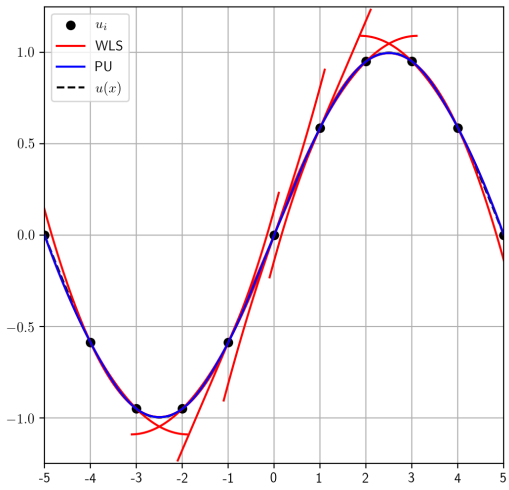


Figure 2.6: PU global approximation to sinusoid constructed by blending local quadratic WLS fits. Constant cloud radius $r = 2.1\Delta x$ is used, corresponding to 5 neighbors on the interior and 3 neighbors on the boundary, including the central node.

2.3 Measuring error

When a global approximation is available, the size of the global error $e^h(\mathbf{x}) = u(\mathbf{x}) - u^h(\mathbf{x})$ can be measured using the p-norms

$$||e^h(\mathbf{x})||_p = \left(\int_\Omega |e^h(\mathbf{x})|^p d\Omega\right)^{1/p}, \tag{2.56}$$

where $u(\mathbf{x})$ is the exact solution and $u^h(\mathbf{x})$ is the global numerical solution. To evaluate the integral, p needs to be chosen and in general quadrature would need to be performed. Common p-norms are the L_1 norm, L_2 norm and L_∞ max-norm

$$||e^h(\mathbf{x})||_1 = \int_\Omega |e^h(\mathbf{x})| d\Omega,$$

$$||e^h(\mathbf{x})||_2 = \sqrt{\int_\Omega |e^h(\mathbf{x})|^2 d\Omega}, \tag{2.57}$$

$$||e^h(\mathbf{x})||_\infty = \max |e^h(\mathbf{x})|$$

When measuring the error for meshfree methods that do not have a global approximation (e.g., GFD), decisions need to be made as to how to discretize the above integrals. More specifically, note that in GFD the numerical solution is available at a set of discrete points as the nodal coefficients $\hat{u}_i$. The nodal coefficients in turn correspond to the overlapping local polynomial approximations. These local solutions are non-interpolatory and are only valid in the neighborhood of their respective point about which they were constructed. This situation is identical to the case of finite differences where the numerical solution is only known at a set of points. However, they differ in that the FD numerical solution corresponds to local solutions that are usually interpolations.

For quasi-uniform meshfree grids, the simplest discretization is to perform nodal quadrature at each node. This results in the corresponding discrete L_1 norm, L_2 norm and L_∞ norm

$$||e^h(\mathbf{x})||_1 = \sum_i |e^h(\mathbf{x}_i)|\Delta V_i,$$

$$||e^h(\mathbf{x})||_2 = \sqrt{\sum_i |e^h(\mathbf{x}_i)|^2 \Delta V_i}, \tag{2.58}$$

$$||e^h(\mathbf{x})||_\infty = \max |e^h(\mathbf{x}_i)|$$

To normalize these norms, $||e^h(\mathbf{x})||_\infty$ can be normalized by max-norm of the solution $||u(\mathbf{x})||_\infty$, $||e^h(\mathbf{x})||_1$ can be normalized by the domain size, to give the measure of the average absolute error in the domain, and $||e^h(\mathbf{x})||_2$ can be normalized by the L_2 norm of the exact solution $||u(\mathbf{x})||_2$. For quasi-uniform meshfree grids often times ΔV_i is approximately constant or assumed to be constant, further simplifying the above expressions. Note the normalized discrete L_2 norm is in fact a popular error metric in SPH, see e.g., Xu et al. [37].

When the grid is non-uniform, the error contribution from each point should not be weighted equally and more accurate quadrature must be performed. However, as stated

earlier, with GFD a global solution is not available and moreover, we do not have a background mesh with appropriate quadrature points. We can construct a global solution by considering the data fitting problem. For example, one possible approach is to fit a second order piecewise linear interpolation through the GFD nodal solution. Such an interpolation requires triangulating the meshfree grid, wherein the nodes will correspond to the vertices of the triangles. The triangulation can then be reused as the background mesh to compute the above integrals by evaluating and summing the contribution from each triangular element as is done in the finite element method.

An alternative approach, is to decouple the construction of the global solution from the mesh and only use the background mesh to evaluate errors. For example, a global solution $u^h(\mathbf{x})$ can be constructed from the discrete solution $u_i^h(\mathbf{x}_i)$ using MLS. Ultimately this amounts to sampling the discrete nodal solution onto the quadrature points of a background mesh using a local weighted least squares procedure about each quadrature point. When evaluating the error on non-uniform grids (see e.g., Section 3.1.3 and Section 3.1.4), we use this approach with a $P2$ reproducing shape functions and degree four Gauss quadrature on triangular elements (2D) or degree three Gauss quadtrature on tetrahedral elements (3D) [83]. Note that with using either of these approaches, care must be taken to ensure that the solution reconstruction procedure does not introduce larger errors that potentially may mask the expected convergence rate.

2.4 Approximate projection method

In this section we detail the approximate projection method, a numerical approach to solving the incompressible Navier-Stokes equations. We will first assume a fixed computational grid. Details and necessary modifications pertaining to moving grids are provided in Section 2.5. Treating each point as a fixed observation point, the incompressible Navier-Stokes

equations are

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{x}} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = \nabla \cdot \sigma \tag{2.59}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2.60}$$

where $\mathbf{x}$ is a fixed spatial point and σ is the stress tensor for a Newtonian fluid with fluid velocity $\mathbf{u}$ and pressure p. To discretize the PDE, we apply point collocation using General Finite Differences. Specifically, spatial derivatives are discretized using the $P2$ weighted least squares shape functions. We arrive at the following non-dimensional system of differential algebraic equations

$$\begin{aligned} \left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{x}} &= -\mathbf{N}(\mathbf{u}) - \mathbf{G}\mathbf{p} + \frac{1}{Re}\mathbf{L}_u\mathbf{u} \\ \mathbf{D}\mathbf{u} &= 0 \end{aligned}, \tag{2.61}$$

where Re is the Reynolds number, $\mathbf{L}_u$ is a block diagonal vector Laplacian and corresponds to the direct discretization of the second derivatives in the viscous terms, $\mathbf{D}$ is the divergence, $\mathbf{G}$ is the gradient, and $\mathbf{N}(\mathbf{u})$ represents the nonlinear convective terms. Applying a mth order backward difference formula (BDF) [84] for the time derivative, taking the right hand side at the $n+1$ time level, and separating boundary terms $\mathbf{b}_1$ and $\mathbf{b}_2$ we have the fully discretized system

$$\begin{aligned} \frac{1}{\Delta t}\left(\alpha_0\mathbf{u}^{n+1} + \sum_{i=1}^{m}\alpha_i\mathbf{u}^{n+1-i}\right) &= -\sum_{i=1}^{r}\beta_i\mathbf{N}\left(\mathbf{u}^{n+1-i}\right) - \mathbf{G}\mathbf{p}^{n+1} + \frac{1}{Re}\mathbf{L}_u\mathbf{u}^{n+1} + \mathbf{b}_1 \\ \mathbf{D}\mathbf{u}^{n+1} &= \mathbf{b}_2 \end{aligned}, \tag{2.62}$$

where the convective term $\mathbf{N}\left(\mathbf{u}^{n+1}\right)$ has been treated explicitly using a rth order extrapolation.

In matrix form we have

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p}^{n+1} \end{bmatrix} = \begin{bmatrix} -\sum_{i=1}^{m} \frac{\alpha_i}{\Delta t} \mathbf{u}^{n+1-i} - \sum_{i=1}^{r} \beta_i \mathbf{N}\left(\mathbf{u}^{n+1-i}\right) + \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \tag{2.63}$$

where $\mathbf{A} = \frac{\alpha_0}{\Delta t}\mathbf{I} - \frac{1}{Re}\mathbf{L}_u$. We will not solve the monolithic equations exactly. Instead, we apply the framework of approximate factorizations to algebraically split the monolithic system into a sequence of cheaper steps [85–88]. An exact block LU factorization of the system reads

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & 0 \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix}. \tag{2.64}$$

To simplify the U step of the factorization, we approximate $\mathbf{A}^{-1}$ in the (1,2) block as $\mathbf{A}^{-1} \approx \frac{\Delta t}{\alpha_0}\mathbf{I}$. To simplify the L step, we approximate the (2,2) block as $\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \approx \frac{\Delta t}{\alpha_0}\mathbf{L}$ leading to the following modified system

$$\begin{bmatrix} \mathbf{A} & 0 \\ \mathbf{D} & -\frac{\Delta t}{\alpha_0}\mathbf{L} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \frac{\Delta t}{\alpha_0}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p}^{n+1} \end{bmatrix} = \begin{bmatrix} -\sum_{i=1}^{m} \frac{\alpha_i}{\Delta t} \mathbf{u}^{n+1-i} - \sum_{i=1}^{r} \beta_i \mathbf{N}\left(\mathbf{u}^{n+1-i}\right) + \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \tag{2.65}$$

where $\mathbf{L}$ is the direct discretization of the second derivatives in the Laplacian ∇^2 with inclusion of homogeneous Neumann boundary conditions on the pressure near solid surfaces. The approximation of the (2,2) block in the L step was chosen to prevent the checkerboarding problem typical of co-located variable arangements [86]. Compared to using $\mathbf{DG}$, the discrete Laplacian $\mathbf{L}$ is cheaper to compute and results in a compact stencil as opposed to what would generally be an extended stencil when variables are co-located. Moreover, a GFD discretization will result in right null space that is comprised only of constant mode

and so there will be no spurious pressure modes. However, note that this discrete GFD Laplacian $\mathbf{L}$ will be asymmetric. Besides breaking symmetry and introducing boundary conditions for pressure, the major consequence of using $\mathbf{L}$ is that the velocity field at $\mathbf{u}^{n+1}$ will not be divergence free, that is the projection is approximate.

More specifically, the algebraic splitting we have chosen introduces splitting errors proportional to the time step and pressure not only in the continuity equation but also in the momentum equation. With the pressure in non-incremental form the momentum equation splitting error is $\frac{\Delta t}{\alpha_0 Re}\mathbf{L}_u\mathbf{G}\mathbf{p}^{n+1}$ whereas the continuity perturbation is $\frac{\Delta t}{\alpha_0}\left(\mathbf{DG}-\mathbf{L}\right)\mathbf{p}^{n+1}$.

The splitting error can be reduced via the time step or by solving for a pressure increment $\Delta\mathbf{p}$ that is lth order accurate [88, 89]. An lth order accurate pressure increment $\Delta\mathbf{p}$ is constructed as

$$\Delta\mathbf{p} = \mathbf{p}^{n+1} - \hat{\mathbf{p}}^{n+1}, \tag{2.66}$$

where $\hat{\mathbf{p}}^{n+1}$ is an lth order extrapolated pressure based on previous pressure values

$$\hat{\mathbf{p}}^{n+1} = \sum_{i=1}^{l} \gamma_i \mathbf{p}^{n+1-i}. \tag{2.67}$$

Eq. 2.65 rewritten with pressure in incremental form is

$$\begin{bmatrix} \mathbf{A} & 0 \\ \mathbf{D} & -\frac{\Delta t}{\alpha_0}\mathbf{L} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \frac{\Delta t}{\alpha_0}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \Delta\mathbf{p} \end{bmatrix} = \begin{bmatrix} -\sum_{i=1}^{m}\frac{\alpha_i}{\Delta t}\mathbf{u}^{n+1-i} - \sum_{i=1}^{r}\beta_i\mathbf{N}\left(\mathbf{u}^{n+1-i}\right) - \mathbf{G}\hat{\mathbf{p}}^{n+1} + \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}. \tag{2.68}$$

Using the general scheme in Eq. 2.68, different schemes can be derived. For BDF-1, a first order semi-implicit Euler scheme is recovered by taking $m = 1, r = 1$ and $l = 0$. The

corresponding coefficients are: $a_0 = 1, a_1 = -1$ and $\beta_1 = 1$. Applying the approximate L and U steps of the factorization we obtain the following sequence of steps

$$\left(\frac{1}{\Delta t}\mathbf{I} - \frac{1}{Re}\mathbf{L}_u\right)\mathbf{u}^* = \frac{1}{\Delta t}\mathbf{u}^n - \mathbf{N}\left(\mathbf{u}^n\right) + \mathbf{b}_1, \tag{2.69}$$

$$\Delta t \mathbf{L}\mathbf{p}^{n+1} = \mathbf{D}\mathbf{u}^* - \mathbf{b}_2, \tag{2.70}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \mathbf{G}\mathbf{p}^{n+1}, \tag{2.71}$$

where $\mathbf{u}^*$ is simply an intermediate storage variable and where note the pressure is in non-incremental form. The first step accounts for the convection and diffusion of fluid momentum, whereas the next two steps, the pressure solve and velocity correction, approximately enforce the divergence-free requirement on velocity.

The above linear systems are solved using either direct (e.g., sparse LU) or iterative solvers (e.g., BiCGSTAB) in the Eigen library [82]. Note that even if the time integration was explicit, the pressure solve would still be implicit due to the nature of the incompressibility requirement. As the pressure solve is often the bottle-neck in incompressible flow calculations, it is advantageous to treat the viscous terms implicitly (as we have done) thereby lifting the diffusive time step restriction ($\Delta t \propto \Delta x^2$). Some incompressible flow solvers try to curtail the need for a pressure solve through an equation of state that relates pressure and density. However, in doing so, these weakly compressible solver introduce Mach number dependence into the solution with density perturbations propagating at approximately the numerical speed of sound. This is in contrast to a truly incompressible solver where the speed of sound is effectively infinite.

When the pressure is not fixed anywhere in the domain (e.g., closed container), the pressure system is singular. To ensure solvability, the discrete solution is constrained to be orthogonal to the constant vector. This is achieved using a single Lagrange multiplier and is equivalent to requiring the average pressure to be zero.

The second scheme BDF-2, is a second order scheme corresponding to $m = 2, r = 2$

and $l = 1$ (incremental pressure). For uniform time steps, the coefficients for this scheme are ($\alpha_0 = 3/2, \alpha_1 = -2, \alpha_2 = 1/2$), ($\beta_1 = 2$, $\beta_2 = -1$), and ($\gamma_1 = 1$). Applying the approximate L and U steps of the factorization we obtain the following similar sequence of steps

$$\left(\frac{3}{2\Delta t}\mathbf{I} - \frac{1}{Re}\mathbf{L}_u\right)\mathbf{u}^* = \frac{1}{\Delta t}\left(2\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}\right) - 2\mathbf{N}\left(\mathbf{u}^n\right) + \mathbf{N}\left(\mathbf{u}^{n-1}\right) - \mathbf{G}\mathbf{p}^n + \mathbf{b}_1, \tag{2.72}$$

$$\frac{2\Delta t}{3}\mathbf{L}\Delta\mathbf{p} = \mathbf{D}\mathbf{u}^* - \mathbf{b}_2, \tag{2.73}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{2\Delta t}{3}\mathbf{G}\Delta\mathbf{p}. \tag{2.74}$$

where note now additional time levels are stored and the pressure increment is solved for as opposed to the pressure itself. With regards to the splitting errors of the BDF-1 and BDF-2 scheme, in Section 4, we measure the convergence rates while choosing a diffusive time step ($\Delta t \propto \Delta x^2$) or convective time step ($\Delta t \propto \Delta x$).

In the following section we detail several approaches that can be used to extend this fixed grid approximate projection method to moving grids.

2.5 Moving meshfree grids

In arbitrary Lagrangian-Eulerian (ALE) methods [28, 90], rather than tracking the state of the flow at a fixed spatial point $\mathbf{x}$ (Eulerian) or a fixed material point $\mathbf{X}$ with a spatial path $\mathbf{x} = \varphi(\mathbf{X}, t)$ (Lagrangian), the state of the flow is tracked along an arbitrary path $\mathbf{x} = \phi(\chi, t)$, where χ are the coordinates identifying a grid point in the reference configuration. By tracking the flow state along an arbitrary path, it is possible to handle moving interfaces while simultaneously maintaining high quality adaptive grids. ALE methods can be classified as either indirect or direct [91].

2.5.1 Indirect (3-stage) ALE methods

In indirect ALE methods, also known as 3-stage ALE methods, the grid nodes are initially treated as particles during the first (Lagrangian) stage. Correspondingly, the grid is evolved by integrating along the particle pathlines

$$\left.\frac{\partial \mathbf{x}}{\partial t}\right|_{\mathbf{x}} = \mathbf{u}, \tag{2.75}$$

where $\mathbf{x}$ and $\mathbf{u}$ are the spatial position and velocity of some material point $\mathbf{X}$ at time t. Neglecting body forces, the momentum equations are solved along the Lagrangian path

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{x}} = \nabla \cdot \sigma, \tag{2.76}$$

subject to the incompressibility constraint $\nabla \cdot \mathbf{u} = 0$. Here, $\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{x}}$ is the acceleration of the particle and $\nabla \cdot \sigma$ is the divergence of the stress tensor for a Newtonian fluid. Note the absence of the convective terms. Ultimately, with appropriate discretization, we obtain the position $\mathbf{x}^{n+1}$ of the particle and its velocity $\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right)$ at the new position. As a pure Lagrangian method can not handle large deformations due to grid degradation, a second stage is introduced wherein the particles are rezoned to maintain a more regular grid

$$\mathbf{x}^s = \mathbf{x}^{n+1} + \Delta\mathbf{x}, \tag{2.77}$$

where $\Delta\mathbf{x}$ is a shift introduced to regularize the grid. Due to this shift, the velocity that was previously solved for no longer corresponds to the new shifted position $\mathbf{x}^s$. To account for this discrepancy, a third and final stage is introduced wherein the velocity field is remapped from the particle configuration at $\mathbf{x}^{n+1}$, to the new shifted position $\mathbf{x}^s$.

While not explicitly stated, many SPH solvers are indeed 3-stage ALE schemes. See for instance the references [7, 35, 37, 38]. Typically, an explicit predictor-correct time integration scheme is applied to solve for the velocity at the Lagrangian position using

either a weakly compressible assumption or applying a projection. The particle is then shifted and a non-conservative remapping strategy is applied. A popular remapping strategy is a Taylor series constructed about the point $\mathbf{x}^{n+1}$

$$\mathbf{u}^{n+1}\left(\mathbf{x}^{s}\right)=\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right)+\nabla\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right)\cdot\left(\mathbf{x}^{s}-\mathbf{x}^{n+1}\right). \tag{2.78}$$

Note the gradient must be constructed consistently and using the configuration $\mathbf{x}^{n+1}$. After the rezoning and remapping stages, the velocity field no longer corresponds to the original material particles. In fact, the new velocity field more closely approximates the velocity field that would be obtained if we were to directly solve the momentum equations while accounting for the relative grid motion with respect to the Lagrangian particle. This is in fact the approach taken in the direct ALE methods which are discussed in Section 2.5.3. Before detailing direct ALE methods, we first briefly discuss another approach [6] motivated by the rezoning and remapping procedures of 3-stage ALE schemes.

2.5.2 Rezoning and remapping Eulerian solutions

Similar to the rezoning and remapping stages of the 3-stage ALE schemes, it is also possible to directly rezone and remap the Eulerian grid solution. This approach is detailed in our previous published work [6] and is the approach used for the results reported on quasi-uniform moving meshfree grids in Section 4.

For the rezoning stage, after solving for the flow field on a fixed grid, the grid nodes are moved along a Lagrangian path. The Lagrangian position is then subsequently shifted using a spring-based approach similar to what is presented later in this thesis. The solution variables at time level $n+1$ from grid configuration at $\mathbf{x}^{n}$ are then non-conservatively remapped to the grid configuration at $\mathbf{x}^{n+1}$. The remapping procedure was tested with a linear approximation $P1$ as well as a quadratic approximation $P2$. Note that the rezoning stage does not have to involve integration along the Lagrangian path, indeed it can also be

along more general paths, although in all examples the former was chosen as it demonstrates the approach is robust with highly deforming grids.

2.5.3 Direct ALE methods

In contrast to 3-stage ALE methods, in direct ALE methods separate rezoning and remapping stages are removed. Instead, they are directly incorporated as follows. First, the grid position is directly integrated along the desired path

$$\left.\frac{\partial \mathbf{x}}{\partial t}\right|_{\chi} = \hat{\mathbf{u}}, \tag{2.79}$$

where $\hat{\mathbf{u}}$ is a specified grid velocity that ensures the grid does not degrade. Second, the fluid velocity is directly solved for at the new grid position $\mathbf{x}^{n+1}$. To accomplish this, the LHS of the momentum equation is modified to account for the relative motion between the grid and the particle. Specifically, the material derivative can be equivalently expressed using the fundamental ALE relation [90]

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\mathbf{x}} = \left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\chi} + (\mathbf{c}\cdot\nabla)\,\mathbf{u}. \tag{2.80}$$

Here, $\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\chi}$ is the rate of change of $\mathbf{u}$ while holding the reference coordinate χ fixed and where

$$\mathbf{c} = \mathbf{u} - \hat{\mathbf{u}} \tag{2.81}$$

is the relative velocity between the material and grid.

Using Eq. 2.80, the ALE formulation of the incompressible Navier-Stokes equations is

$$\begin{aligned} \left.\frac{\partial \mathbf{x}}{\partial t}\right|_{\chi} &= \hat{\mathbf{u}} \\ \left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\chi} + (\mathbf{c}\cdot\nabla)\,\mathbf{u} &= \nabla\cdot\sigma \\ \nabla\cdot\mathbf{u} &= 0, \end{aligned} \tag{2.82}$$

subject to appropriate initial conditions and boundary conditions.

In the momentum equation, the choice of the ALE convective velocity $\mathbf{c}$ can be broken down into three distinct categories which are summarized below

1. $\mathbf{c} = \mathbf{0}$: Choosing $\hat{\mathbf{u}} = \mathbf{u}$, the grid points will evolve along the particle path lines $\mathbf{x} = \varphi(\mathbf{X}, t)$ and so Eqs. 2.82 will result in a purely Lagrangian method.

2. $\mathbf{c} = \mathbf{u}$: Choosing $\hat{\mathbf{u}} = 0$, the grid points will be fixed observation points and so Eqs. 2.82 will recover the Eulerian formulation given by Eq. 2.60.

3. $\mathbf{c} \neq \mathbf{0}$ and $\mathbf{c} \neq \mathbf{u}$: Choosing a non-zero grid velocity where $\hat{\mathbf{u}} \neq \mathbf{u}$ will result in grid points following an arbitrary path $\mathbf{x} = \phi(\chi, t)$. This choice results in the general ALE formulation given by Eq. 2.82.

With respect to the semi-implicit approximate projection solver detailed in Section 2.4, only a few changes need to be made to extend it to handle Eq. 2.82. For now, we will assume we have some approach to construct a grid velocity $\hat{\mathbf{u}}$ that maintains a regular adaptive grid. Exactly how to construct the grid velocity will be detailed subsequently in Section 2.5.4.

Focusing on the momentum equation, if we want to re-use the implicit BDF-k temporal schemes, the LHS and RHS will need to be evaluated at time level $n+1$. As before, this includes the time derivative, the convective terms, and the divergence of the stress tensor. Contrary to before, the grid position at time level $n+1$ is now controlled by the grid

evolution equation. As such, to obtain the spatial operators at the new grid position, first the grid position needs to be integrated explicitly, e.g.

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \Delta t \hat{\mathbf{u}}, \tag{2.83}$$

Second, note the temporal discretization is now along the grid path. As such, when performing integration, the fluid velocity at the different time levels will also correspond to different positions along the grid path. For instance, a BDF-1 temporal discretization will give

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\chi}^{n+1} = \frac{1}{\Delta t}\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right) - \frac{1}{\Delta t}\mathbf{u}^{n}\left(\mathbf{x}^{n}\right), \tag{2.84}$$

whereas a BDF-2 temporal discretization will give

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\chi}^{n+1} = \frac{3}{2\Delta t}\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right) - \frac{2}{\Delta t}\mathbf{u}^{n}\left(\mathbf{x}^{n}\right) + \frac{1}{\Delta t}\mathbf{u}^{n-1}\left(\mathbf{x}^{n-1}\right). \tag{2.85}$$

Furthermore, as we would like to avoid non-linear iterations, the implicit non-linear term $\left(\mathbf{c}^{n+1}\cdot\nabla\right)\mathbf{u}^{n+1}$ needs to be extrapolated. Whereas the Eulerian formulation leads to an extrapolation in time at a fixed spatial point, in the ALE formulation this term must be extrapolated using a space-time Taylor series as both the position and time level of the nodes are changing. The simplest approximation is a first order space-time approximation

$$\left(\mathbf{c}^{n+1}\cdot\nabla\right)\mathbf{u}^{n+1} \approx \left(\mathbf{c}^{n}\cdot\nabla\right)\mathbf{u}^{n}, \tag{2.86}$$

where all terms are computed using the previous grid configuration. This approximation results in first order temporal errors as well as first order spatial errors.

To remove the first order spatial error, a more accurate representation of the $\mathbf{c}^{n+1}$ must be constructed. For the case of rigid body motion, it is possible to construct a known grid velocity at the new time level $\hat{\mathbf{u}}^{n+1}$ that is entirely decoupled from the fluid flow (see Section 2.5.4). As such, if we treat the gradient term implicitly, only the unknown fluid

velocity $\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right)$ in $\mathbf{c}^{n+1} = \mathbf{u}^{n+1} - \hat{\mathbf{u}}^{n+1}$ needs to be approximated. This is no trivial task as it requires a linear space-time Taylor series. Alternatively, a quicker route to an equivalent approximation is to explicitly integrate the fundamental ALE relation

$$\mathbf{u}^{n+1}\left(\mathbf{x}^{n+1}\right) \approx \mathbf{u}^{n}\left(\mathbf{x}^{n}\right) + \Delta t\left(\mathbf{a}^{n}\left(\mathbf{x}^{n}\right) - \nabla\mathbf{u}^{n}\cdot\left(\mathbf{u}^{n}\left(\mathbf{x}^{n}\right) - \hat{\mathbf{u}}^{n}\left(\mathbf{x}^{n}\right)\right)\right), \tag{2.87}$$

where $\mathbf{a}^{n}\left(\mathbf{x}^{n}\right)$ is the acceleration of the fluid particle passing through point $\mathbf{x}^{n}$ and where the gradient term is also constructed at the previous grid configuration. In Chapter 5, we present results using the direct ALE method with the grid velocity constructed as described next in Section 2.5.4.

2.5.4 Grid velocity construction

Construction of the grid velocity is key to the success of ALE methods as it directly controls how the grid evolves in time. While there are many possible strategies, we choose to construct the grid velocity $\hat{\mathbf{u}}$ as a combination of two components

$$\hat{\mathbf{u}} = \hat{\mathbf{u}}_1 + \hat{\mathbf{u}}_2, \tag{2.88}$$

where $\hat{\mathbf{u}}_1$ is a smooth component and where $\hat{\mathbf{u}}_2$ is a grid regularization component defined as

$$\hat{\mathbf{u}}_2 = \Delta\mathbf{x}_s/\Delta t, \tag{2.89}$$

where $\Delta\mathbf{x}_s$ is a regularizing shift constructed to ensure 1) the grid does not deteriorate and 2) higher resolutions are maintained in areas of interest. As detailed in Section 2.6, to construct the shift we use Eq. 2.93. This shift is computed explicitly and corresponds to relaxing a repulsive network of linear springs with (potentially) varying equilibrium spring lengths. By specifying the equilibrium spring lengths as a function of space and time, the grid spacing and adaptivity can be controlled throughout the domain as well as throughout

time.

To construct the smooth component $\hat{\mathbf{u}}_1$ we consider two approaches. The first approach is to set the smooth component equal to the fluid velocity $\hat{\mathbf{u}}_1 = \mathbf{u}$. With this smooth component, the direct ALE scheme is analogous to the 3-stage ALE schemes used commonly in the SPH community. However, now the rezoning stage is directly merged into the integration of the grid position, whereas the remapping stage is directly merged in the momentum solve. The advantages of using $\hat{\mathbf{u}}_1 = \mathbf{u}$ include 1) there are no costs to constructing it, 2) the non-linear ALE convective term has less of an influence as $\mathbf{c} \approx \Delta\mathbf{x}_s/\Delta t$ should be small in comparison to an Eulerian solver where $\mathbf{c} = \mathbf{u}$ and 3) using $\mathbf{u}$ for the smooth component naturally handles the problem of moving grid points around moving obstacles. The main disadvantage of this approach is the grid motion is coupled to the fluid flow even for the case where boundary kinematics are known ahead of time. As we show in later sections, for large Reynolds numbers the grid will undergo significant deformations. While the grid regularity can still be maintained through successive application of the regularizing component, from a computational perspective, large grid deformations are never ideal.

A second approach is to construct the smooth grid velocity from the solution to the Laplace equation [28]

$$\nabla^2 \hat{\mathbf{u}}_1 = 0, \tag{2.90}$$

subject to $\hat{\mathbf{u}}_1|_{\Gamma_1} = \mathbf{u}$ on the moving boundary Γ_1 and $\hat{\mathbf{u}}_1|_{\Gamma_2} = 0$ on the non-moving boundary Γ_2. In this manner, moving boundaries may be tracked using Lagrangian particles whereas stationary boundaries will be discretized using fixed Eulerian points that allow material to flow through them if need be (e.g., inlet or outlet). For prescribed rigid body motion, this approach is particularly advantageous as the grid motion is entirely decoupled from the fluid flow and only depends on the kinematics of the boundaries. One of the side benefits of decoupling the grid motion from the fluid flow is that it is possible to perform a dry run and only solve for the grid motion throughout the integration period. During this dry run the grid can then be inspected and visualized prior to solving for the fluid flow.

In Chapter 5, a comparison of first two approaches is detailed for the case of an inline oscillating cylinder with grid adaptivity near the surface.

A third approach, although tested but not shown in this work, is to construct the smooth component according to potential flow. Specifically, assuming an irrotational smooth component, $\hat{\mathbf{u}}_1 = \nabla\phi$, the velocity potential ϕ may be solved for by imposing the divergence free condition. This results in the Laplace equation for the velocity potential

$$\nabla^2\phi = 0, \tag{2.91}$$

subject to $\hat{\mathbf{u}}_1 \cdot \hat{n} = \nabla\phi \cdot \hat{n}$ on the boundaries. With this approach, the grid motion 1) depends only on the kinematics of the boundaries and 2) will slip around moving objects. A potential advantage of this approach is that only a single scalar needs to be solved.

2.6 Meshfree grid generation, regularization, and adaptivity

Contrary to what the name might imply, meshfree methods are not free from the troubles of grid generation and grid regularity (quality), two topics which have received much attention from mesh-based methods. In this section, we illustrate a simple spring-based approach that can be used to tackle meshfree grid generation, grid regularity and grid adaptivity.

Specifically, we extend Persson and Strang's explicit mesh-based grid generation approach [78] to meshfree grids. In their approach, an initial set of the grid nodes is evolved according to the relaxation equation

$$d\mathbf{x}/d\tau = \mathbf{F}\left(\mathbf{x}\right), \tag{2.92}$$

where $\mathbf{F}$ is the net spring force acting on a given node. At equilibrium, the distance between neighboring nodes is approximately the local spring equilibrium length with $\mathbf{F}\left(\mathbf{x}\right) = 0$.

In the meshfree version presented here, rather than using the edges in a Delanauy triangulation, springs are allowed to overlap and are defined as the segments that connect a

given node to its neighbors. That is, the Delanauy triangulation is removed and is replaced with a k nearest neighbor search. Applying a forward Euler discretization to the relaxation equation and assuming linear springs, we obtain the following explicit shift $\Delta \mathbf{x}_i$ for node i in the network

$$\Delta \mathbf{x}_i = -\Delta\tau \sum_j^n \mathbf{x}_{ji} \max\left(\frac{r_{eq}\left(\mathbf{x}_m\right)}{||\mathbf{x}_{ji}||} - 1, 0\right). \tag{2.93}$$

Here, $\Delta\tau$ is the relaxation parameter, $\mathbf{x}_{ji} = \mathbf{x}_j - \mathbf{x}_i$ is the ray going from a grid node i to its jth neighbor, $||\mathbf{x}_{ji}||$ is the length of the spring between the nodes, and r_{eq} is the relative equilibrium spring length function evaluated at the midpoint $\mathbf{x}_m = 0.5\left(\mathbf{x}_i + \mathbf{x}_j\right)$. In 2D, we take the summation only over the $n = 7$ nearest neighbors (including the central node).

When generating grids, the relaxation given by Eq. 2.93 is used to iterate not only on the interior nodes but also on the boundary points as well. Points are allowed to equilibrate on the boundary but they are not allowed to cross normal to the boundary. To enforce these boundary conditions a normal reaction force is applied. Specifically, if a point penetrates the domain boundary it is projected back onto the surface using the surface normal $\hat{\mathbf{n}}$

$$\mathbf{x}^* = \mathbf{x} - d_\Gamma\left(\mathbf{x}\right)\hat{\mathbf{n}}, \tag{2.94}$$

where, $d_\Gamma\left(\mathbf{x}\right)$ is the signed distance function with the zero level set $d_\Gamma = 0$ defining the domain boundary, $d_\Gamma > 0$ the exterior, and $d_\Gamma < 0$ the interior. Rather than carry out the exact differentiation, the surface unit normal can be estimated using a small epsilon in a finite difference approximation to the gradient of the signed distance function $\nabla d_\Gamma\left(\mathbf{x}\right)$. Representing the domain boundary in this manner is particular attractive as complex shapes can easily be represented as the zero level set.

In the relaxation given by Eq. 2.93, the spring equilibrium length function $r_{eq}\left(\mathbf{x}\right)$, directly controls the spacing between grid points at equilibrium. Consequently, different choices for the equilibrium spring length function result in different adaptive grid strate-

gies. A simple strategy is to impose geometric adaptivity via an additional separate signed distance function. Specifically, the spring equilibrium length r_{eq} can be computed as a linear interpolation

$$r_{eq}(\mathbf{x}) = h_0 \left(1 + \frac{d_a(\mathbf{x})}{d_{a,\min}} \left(\frac{\Delta x_{max}}{\Delta x_{min}} - 1\right)\right), \tag{2.95}$$

where $d_a(\mathbf{x})$ is the signed distance function used for grid adaptation, and $d_{a,\min}$ is the minimum of this signed distance function and ensures the latter quantity $d_a(\mathbf{x})/d_{a,\min} \in [0,1]$, and where $\Delta x_{max}/\Delta x_{min}$ is the specified clustering ratio. Furthermore, h_0 is directly proportional to the minimum spacing ($i.e., h_0 \approx 1.2\Delta x_{min}$). Taking two examples, if the point $\mathbf{x}$ corresponds to the farthest distance from the boundary, then the equilibrium spring length will be $r_{eq} \approx h_0 \Delta x_{max}/\Delta x_{min}$. Instead, if $d_a(\mathbf{x}) = 0$ then $r_{eq} \approx h_0$. The spring equilibrium length $r_{eq}(\mathbf{x})$ can also be defined based on other measures (e.g., curvature, local feature size, truncation errors and derivatives, etc). We do not consider these options in this work and leave it to future work.

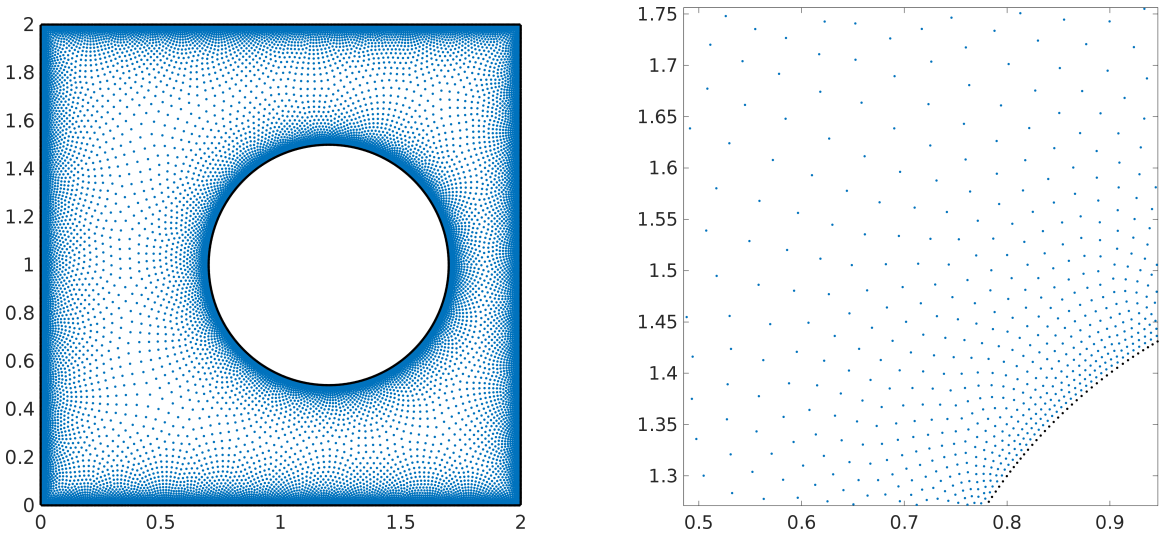
2.6.1 Example 1

Figure 2.7 illustrates an example grid generated using this approach for the oscillating cylinder benchmark presented in Section 5.1. Details regarding the grid initialization procedure have been left off here but can be found in the publication [78]. For this example, $r_{eq}(\mathbf{x})$ is constructed using a signed distance function that (isotropically) resolves the viscous boundary layer

$$d_a(\mathbf{x}) = d_\Gamma(\mathbf{x}) = \max\left(d_1(\mathbf{x}), -d_2(\mathbf{x})\right), \tag{2.96}$$

where d_1 is the approximate signed distance function to the four rectangular walls and $d_2(\mathbf{x}) = ||\mathbf{x} - \mathbf{x}_c|| - R$ is the signed distance function for the cylinder with center $\mathbf{x}_c$ and radius R. Note that the adaptive signed distance function defines the spring equilibrium

length both on the boundary and on the interior and in this example coincides with the boundary domain (i.e., $d_a = d_\Gamma$). This need not be the case. In Section 3.1.4, we show a case where the two do not coincide (i.e., $d_a \neq d_\Gamma$)).



(a) Zoomed out.

(b) Zoomed in around cylinder surface.

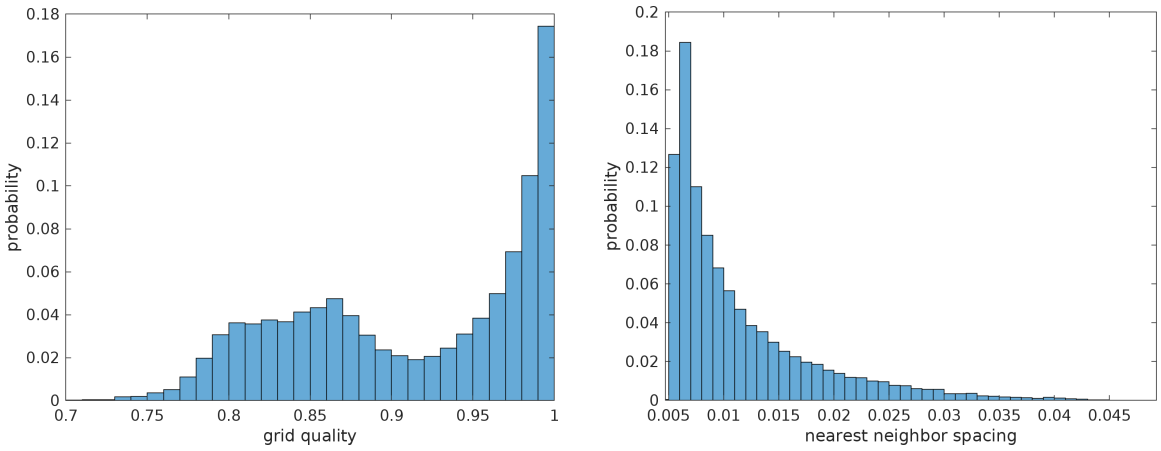
Figure 2.7: Example grid for oscillating cylinder with $\Delta x_{max}/\Delta x_{min} = 10$ and 16358 total interior grid nodes.

After iterating, the spring forces spread out the initial distribution of points to the high quality distribution of points shown. Figure 2.8 plots the grid quality and binned nearest neighbor distance for the grid configuration shown in Figure 2.7. As can be seen, the final grid adheres to the specified clustering ratio with $\Delta x_{max}/\Delta x_{min} \approx 10$ and initial lattice resolution $\Delta x_{min} \approx 1/200$. We are not aware of a standard measure of grid quality for meshfree methods and as such have chosen to report a standard mesh-based quality measure. Specifically, we compute the ratio of two times the radius of the largest inscribed circle to the radius of the smallest circumscribed circle

$$q = \frac{(b+c-a)\,(c+a-b)\,(a+b-c)}{abc}, \tag{2.97}$$

where a, b, c are the side lengths of the triangles in a Delanauy triangulation of the grid points [78]. When $q = 1$, the triangle is equilateral and when $q = 0$, the triangle is

degenerate (i.e., zero area). For the configuration shown, the average mesh-based grid quality $q_{avg} = 0.91$ indicating the grid is of high quality due to the high percentage of near equilateral angles.



(a) Grid quality distribution.

(b) Resolution distribution.

Figure 2.8: Grid quality q and resolution distributions for the grid generated in Figure 2.7.

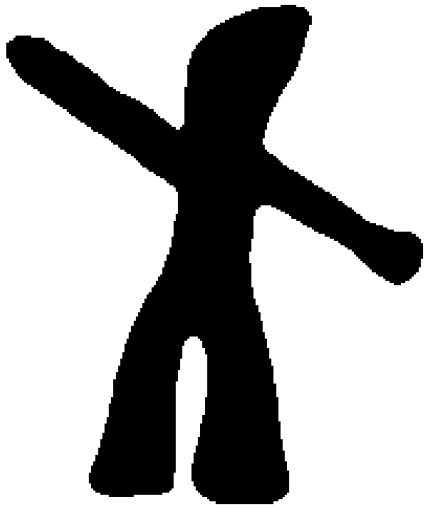
At the four corners, control points were specified to account for the inaccurate projections that result when the approximate signed distance function to the four walls is used. In 3D, the intersection of the boundaries results in control lines (2 surfaces) and control points (3 surfaces). While specifying one or two control points is trivial, specifying a control line becomes problematic as the point density along line should conform to the desired equilibrium length. To alleviate the burden of specifying control lines and points, the correct signed distance function must be specified.

While we have focused the discussion on generating the initial meshfree grid. It is straightforward to introduce time dependence. For this example, if the cylinder is prescribed rigid body motion then $d_2(\mathbf{x}, t) = ||\mathbf{x} - \mathbf{x}_c(t)|| - R$ and so the spring equilibrium length will also be a function of time $r_{eq}(\mathbf{x}, t)$ as well as the zero level set $d_\Gamma(\mathbf{x}, t) = 0$. As the domain boundaries evolve, the relaxation given by Eq. 2.93 is re-computed. The regularizing shift can then be applied to the nodes with the resulting spring-based velocity $\Delta x/\Delta t$ used in the direct ALE solver. In this manner, distance-based adaptivity and grid

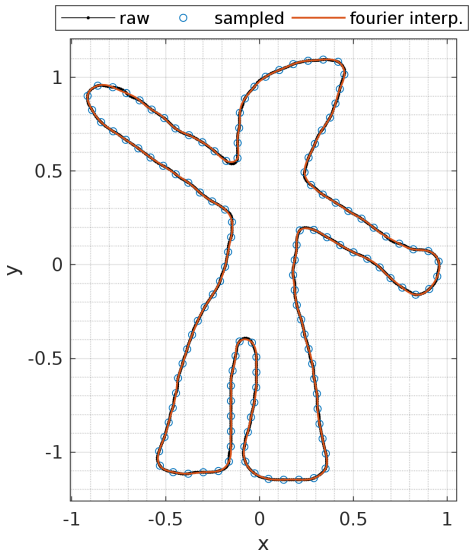
regularity can be maintained as the object undergoes rigid body motion.

2.6.2 Example 2

In the previous example, we defined the domain $d_\Gamma(\mathbf{x})$ by using logic operations on explicit signed distance functions that represent simpler domains (i.e., a rectangle and a circle). However, more often than not, the explicit signed distance function is not known for a desired boundary. In this case, the boundary may be represented as a parametric curve (surface), an implicit function, via a triangulation, or some other way.



(a) Black and white image of a desired boundary.

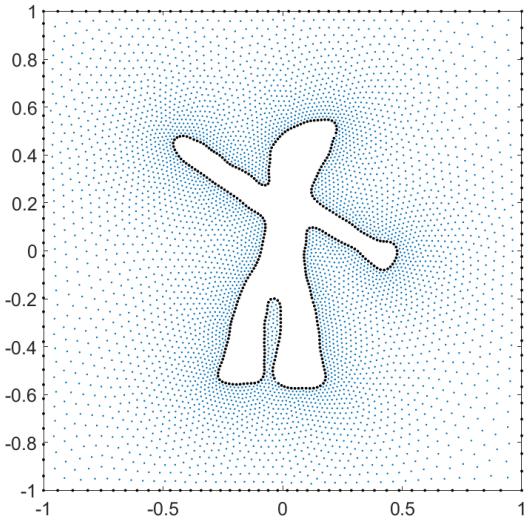


(b) Fourier interpolation of the boundary.

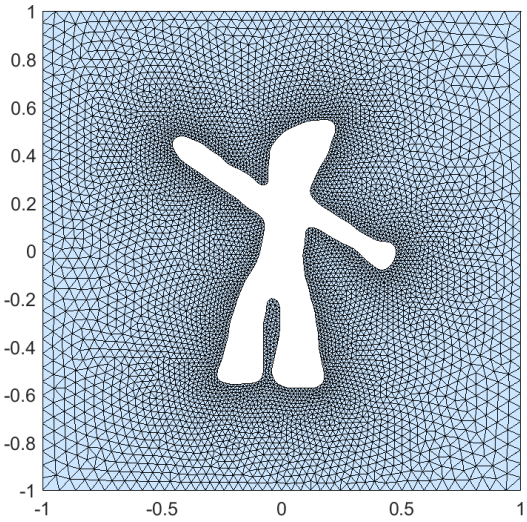
Figure 2.9: Example of a complex geometry.

If the boundary can be represented as an implicit function or parametric curve (surface), Newton iterations can be used to compute the minimum distance to the curve (surface) if needed. If only a boundary representation is needed, (potentially) expensive Netwon iterations may be entirely avoided by using a cheap first order approximate projection [92] to approximately project points back onto the surface. If this is the case, then only the gradient of the implicit function needs to be computed. For instance, the quasi-uniform

grids in the leaflet example in Section 3.1.2 were constructed using this approach.



(a) Resulting grid using the meshfree approach.

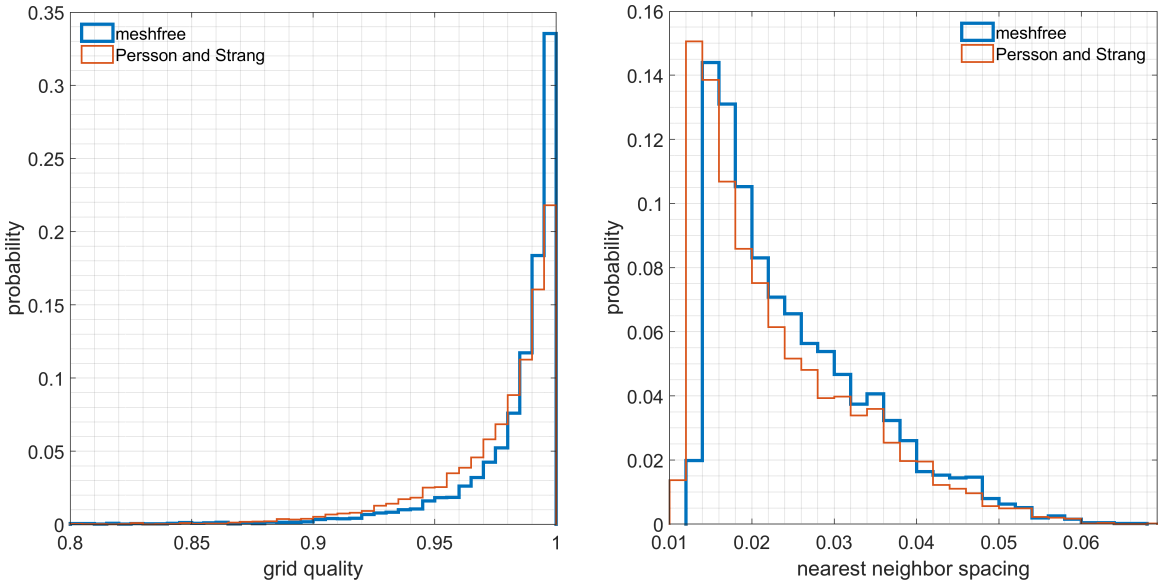


(b) Resulting grid using Persson's Delanauy-based approach.

Figure 2.10: Qualitative comparison between the grids generated using our meshfree grid generator versus Persson's mesh-based approached. The grid is adapted based on the distance to the curve.

In Figure 2.9, we illustrate a case with a more complex geometry. For this example, we construct a smooth parametric curve through chosen boundary points using Fourier descriptors. The resulting boundary interpolation is shown in Figure 2.9b. Once this parametric representation is obtained, we can construct the signed distance function at any point to the curve by solving the distance minimization problem. The signed distance function can then be used to help construct the domain $d_\Gamma(\mathbf{x})$. After specifying the desired spring equilibrium length, a grid can be generated. In Figure 2.10a, we show an example grid constructed with adaptivity based on the distance to the cartoon surface.

Using the same definition of the domain and adaptivity, in Figure 2.10b, we illustrate the resulting grid generated using Persson's mesh-based MATLAB code that is available online. Qualitatively, we see the meshfree and mesh-based approaches generate similar grids. In Figure 2.11, we quantify how well the two grids agree by plotting the grid quality and binned nearest neighbor distance for each grid. As can be seen, both grids generate similar distributions for the both the quality and binned nearest neighbor distances.



(a) Grid quality distribution.

(b) Resolution distribution.

Figure 2.11: Quantitative comparison of grid quality q and resolution distributions for the grids shown in Figure 2.10.

CHAPTER 3

STATIONARY MESHFREE GRIDS

3.1 Elliptic equation

In this section, we apply GFD to solve the Poisson equation in 2D and 3D,

$$-\nabla \cdot k\nabla T = q, \tag{3.1}$$

where q is a heat source (or sink), $\nabla \cdot \nabla$ is the Laplacian operator, and k is the thermal conductivity. In this section, we assume a unity conductivity $k = 1$. Note that the Laplacian operator appears not only in the viscous terms in incompressible flow but also appears when solving for a pressure that enforces the incompressibility constraint. Consequently, the above model equation is important to study.

We start by solving Eq. 3.1 on a simple 2D unit square domain. Using $P2$, $P3$ and $P4$ shape functions we present a detailed truncation error analysis that illuminates the observed convergence rates for the solution error. Subsequently, in Section 3.1.1, we revisit the same manufactured problem to study error estimation and correction through the deferred correction approach. Using this simple example, we illustrate how the $P2$ stencil - with only $n = 13$ neighbors - can be corrected from second order to fourth order as well as how accurate estimates of the error can be constructed.

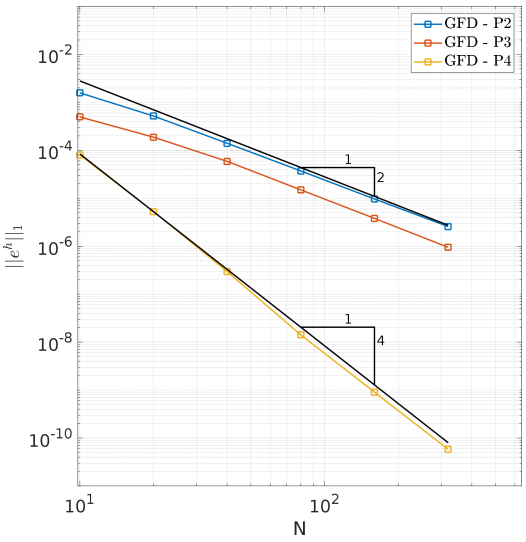
We continue with a study involving complex geometry, verifying that the expected convergence rates hold with and without deferred corrections, as well as again carry out another error estimation analysis. Thereafter, in Section 3.1.3, we study h-adaptivity around a discontinuity in the well known L-shape problem. We conclude in Section 3.1.4, with a 3D h-adaptivity study on a quarter sphere to highlight the flexibility of the grid generation approach.

3.1.1 Square domain

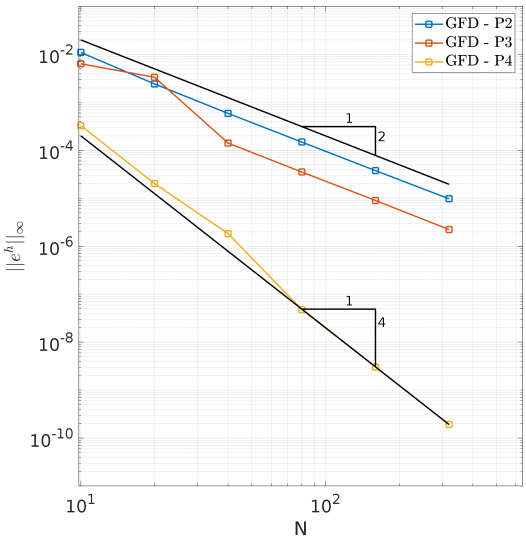
We consider the 2D Poisson equation on a unit square domain. We impose the following analytic solution

$$T(x,y) = \left(1-x^2\right)\left(2y^3 - 3y^2 + 1\right). \tag{3.2}$$

We are free to subject the boundaries to either Neumann boundary conditions and or Dirichlet boundary conditions. Here, we focus on the case wherein appropriate Dirichlet boundary conditions are imposed on the left and right boundaries while Neumann conditions are enforced on the top and bottom boundaries.



(a) Average absolute error for $P2$, $P3$, and $P4$ approximations.



(b) Max pointwise error for $P2$, $P3$, and $P4$ approximations.

Figure 3.1: Convergence rate when Dirichlet boundary conditions are imposed on left and right and Neumann on top and bottom without using deferred corrections.

In Figure 3.1, we plot the convergence rate for quasi-uniform meshfree grids corresponding to $N \approx 10$ to $N \approx 320$ points per side. We see that the $P2$ and $P3$ reproducing WLS shape functions give second order convergence rates for the solution error in the $||\cdot||_1$ norm, whereas $P4$ reproducing WLS shape functions result in a fourth order convergence rate. The number of neighbors used here correspond to a fixed $n = 13$, 21, and 40 for $P2$, $P3$ and $P4$ approximations, respectively. Similar convergence rates are observed in

Figure 3.1b where instead the maximum pointwise error is plotted. Below we try to explain the observed convergence rates by using a truncation error analysis.

Unlike FD truncation error analysis, our grid is not structured and is instead quasi-uniform. As such, in general the stencil coefficients will vary from point to point. Using a multidimensional Taylor series, the truncation errors for the three approximations are summarized below where $\mathbf{x}_i$ is either an interior point or a point on a Neumann boundary

$$P2: \begin{cases} \tau_i \approx \frac{1}{3!}\nabla^3 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i \approx \frac{1}{3!}\nabla^3 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i, & i \in \Gamma_n \end{cases} \tag{3.3}$$

$$P3: \begin{cases} \tau_i \approx \frac{1}{4!}\nabla^4 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i \approx \frac{1}{4!}\nabla^4 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i, & i \in \Gamma_n \end{cases} \tag{3.4}$$

$$P4: \begin{cases} \tau_i = \frac{1}{5!}\nabla^5 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i = \frac{1}{5!}\nabla^5 T|_{\mathbf{x}_i} \vdots \sum_j \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \otimes \mathbf{x}_{ji} \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i, & i \in \Gamma_n \end{cases} \tag{3.5}$$

where $\mathbf{x}_{ji} = \mathbf{x}_j - \mathbf{x}_i$, $\nabla^n T|_{\mathbf{x}_i}$ indicates the sequential application of the gradient operator to the exact solution and results in a rank n tensor comprised of nth order partial derivatives, $\otimes$ is the tensor product, and the vertical dots indicate tensor contraction (e.g., $\vdots$ is the inner product of rank 3 tensors which is a scalar). The shape functions ϕ_j vary between the expressions and will correspond to the $P2$, $P3$, and $P4$ reproducing WLS shape functions,

respectively. Using the analytic solution, the simplified truncation errors are

$$P2: \begin{cases} \tau_i \approx \frac{1}{3!} \sum_j \left(3T_{xxy}|_{\mathbf{x}_i} x_{ji}^2 y_{ji} + 3T_{xyy}|_{\mathbf{x}_i} x_{ji} y_{ji}^2 + T_{yyy}|_{\mathbf{x}_i} y_{ji}^3\right) \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i \approx \frac{1}{3!} \sum_j \left(3T_{xxy}|_{\mathbf{x}_i} x_{ji}^2 y_{ji} + 3T_{xyy}|_{\mathbf{x}_i} x_{ji} y_{ji}^2 + T_{yyy}|_{\mathbf{x}_i} y_{ji}^3\right) \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i, & i \in \Gamma_n \end{cases} \tag{3.6}$$

$$P3: \begin{cases} \tau_i \approx \frac{1}{4!} \sum_j \left(6T_{xxyy}|_{\mathbf{x}_i} x_{ji}^2 y_{ji}^2 + 4T_{xyyy}|_{\mathbf{x}_i} x_{ji} y_{ji}^3\right) \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i \approx \frac{1}{4!} \sum_j \left(6T_{xxyy}|_{\mathbf{x}_i} x_{ji}^2 y_{ji}^2 + 4T_{xyyy}|_{\mathbf{x}_i} x_{ji} y_{ji}^3\right) \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i, & i \in \Gamma_n \end{cases} \tag{3.7}$$

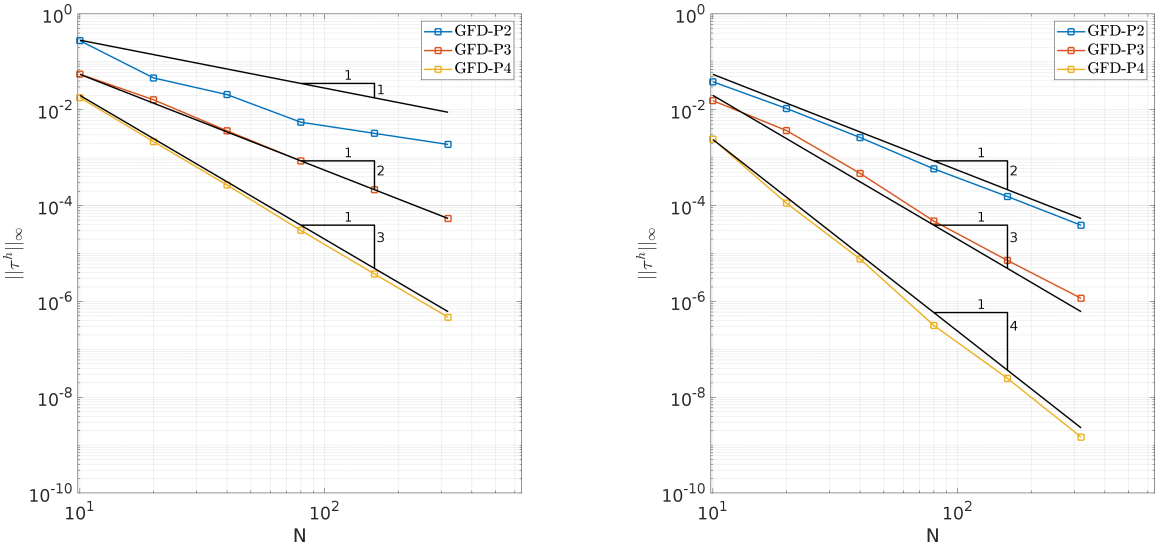
$$P4: \begin{cases} \tau_i = -2 \sum_j x_{ji}^2 y_{ji}^3 \nabla \cdot \nabla \phi_j|_{\mathbf{x}_i}, & i \in \Omega - \Gamma_n \\ \tau_i = -2 \sum_j x_{ji}^2 y_{ji}^3 \nabla \phi_j|_{\mathbf{x}_i} \cdot \hat{n}_i & i \in \Gamma_n \end{cases} \tag{3.8}$$

where for $P4$, the truncation error is exact as there are no higher derivatives than 5th order in the analytic solution.

In Figure 3.2, we measure the convergence rate of the leading truncation errors given by Eq. 3.6 - Eq. 3.8 using the $||\cdot||_\infty$ norm. Derivatives are computed using the exact analytic solution. It is observed that max leading truncation errors are respectively $O(h)$, $O(h^2)$, and $O(h^3)$ for $P2$, $P3$ and $P4$ for interior points (Figure 3.2a) and one order higher for the boundary points with Neumann conditions (Figure 3.2b).

Looking more closely, Figure 3.2a shows that $||\tau^h||_\infty$ for $P2$ is somewhere between $O(h)$ and $O(h^2)$. For these quasi-uniform grids, we observed that the $P2$ shape functions with $n = 13$ neighbors satisfy the positivity conditions $\nabla \cdot \nabla \phi_j|_{\mathbf{x}_i} > 0$ for neighboring nodes. As such, we expect on average the odd moments in the leading truncation term will cancel out for points away from the boundary if these points also have symmetric stencils coefficients. However, it is apparent that some of the quasi-uniform meshfree grids have

rogue points with asymmetric stencils (especially near corners). Consequently, the odd moments in the interior truncation error term do not cancel out.



(a) Max pointwise truncation error for $P2$, $P3$, and $P4$ approximations for $\mathbf{x}_i$ in $\Omega - \Gamma_n$.

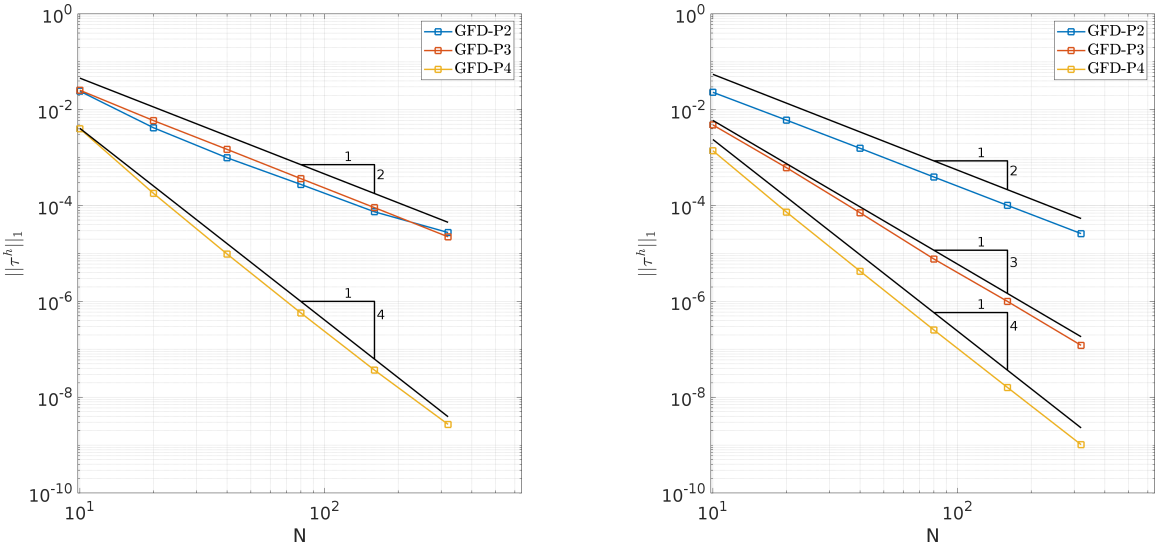
(b) Max pointwise truncation error for $P2$, $P3$, and $P4$ approximations for $\mathbf{x}_i$ in Γ_n.

Figure 3.2: Convergence rate of the max pointwise truncation error τ as computed using Eqs. 3.6 - 3.8 for interior points (left) and boundary points (right).

In Figure 3.3a, we plot the leading truncation error using the $||\cdot||_1$ norm for the interior for all three approximations. We see that the average absolute pointwise leading truncation error converges similarly to the rates reported for the solution error in Figure 3.1. Specifically, the $||\tau^h||_1$ convergence rate for $P2$ nearly matches the $||\tau^h||_1$ convergence rates for $P3$ and eventually tapers off for small enough h. It appears that on average, the stencils are indeed well balanced and the partial cancellation of odd moments in the $P2$ Laplacian truncation term will occur if $n = 13$ neighbors are used.

Similar behavior is observed for the $P4$ approximation, where $||\tau^h||_1$ error converges at an increased order as compared to $||\tau^h||_\infty$. Again, the $||\cdot||_1$ measure is inline with the measured solution error convergence rates. Surprisingly, this is despite the fact that we found the $P4$ stencils violate the positivity conditions. On the other hand, $||\tau^h||_1$ error for $P3$ is second order for the interior as there is an even moment in the leading term that will not cancel. Lastly, note that none of the approximations have cancellation in the truncation

error of the Neumann boundary conditions as can be seen in Figure 3.3b. These results suggest that the solution error converges in line with the convergence rates of the leading truncation error term in the $||\cdot||_1$ norm rather than $||\cdot||_\infty$ norm.



(a) Average absolute pointwise truncation error for $P2$, $P3$, and $P4$ approximations for $\mathbf{x}_i$ in the interior $\Omega - \Gamma_n$.

(b) Average absolute pointwise truncation error for $P2$, $P3$, and $P4$ approximations for $\mathbf{x}_i$ on the Neumann boundaries Γ_n.

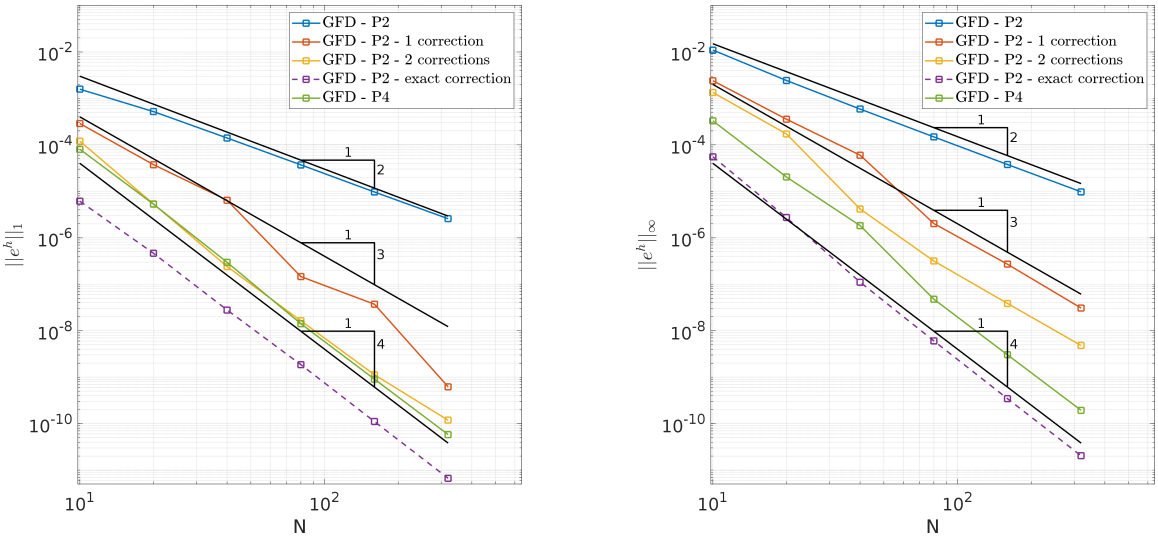
Figure 3.3: Convergence rate of the average absolute truncation error τ as computed using Eqs. 3.6 - 3.8 for interior points (left) and boundary points (right).

Deferred correction and error estimation

As stated earlier, the $P4$ truncation error given in Eq. 3.8 is exact. Indeed, if we apply the deferred correction given by Eq. 2.24 with the truncation error specified by Eq. 3.8, we will recover the exact pointwise solution to machine precision for the $P4$ discretization. This is assuming we use the exact analytic solution to compute the derivatives.

Similarly, we could make the $P2$ approximation exact by adding back the missing higher moments. However, for $P2$, besides the leading truncation error given by Eq. 3.6, the higher order terms given by Eq. 3.7 and Eq. 3.8 must be included, where the shape functions used in the calculation should correspond to the $P2$ shape functions as opposed to $P3$ and $P4$. Similarly, to make $P3$ exact, besides Eq. 3.7, the truncation term in Eq. 3.8 would need to be computed using the $P3$ shape functions.

At first glance, it may seem we are in a dilemma. To correct the $P2$ approximation to be fourth order accurate, we need to estimate third and fourth order partial derivatives. This is no trivial matter and to be consistent would require the expensive $P4$ shape functions. Moreover, the estimate of the required partial derivatives will be constructed from data that is lower order which may compromise our estimates. With this said, we note that the stencil in $\mathbf{L}^h$ will not change and will correspond to the cheaper $P2$ stencil. Consequently, the expense of computing the right hand side potentially may be justified.



(a) Average absolute error for $P2$ approximation with exact and approximate deferred corrections compared to the $P4$ approximation.

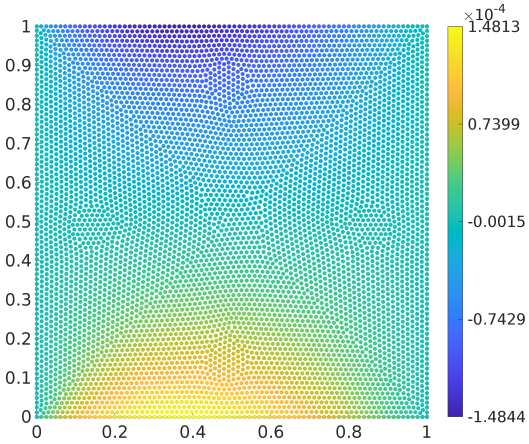
(b) Max pointwise error for $P2$ approximation with exact and approximate deferred corrections compared to the $P4$ approximation.

Figure 3.4: Convergence rate when Dirichlet boundary conditions are imposed on left and right and Neumann on top and bottom when using deferred corrections.

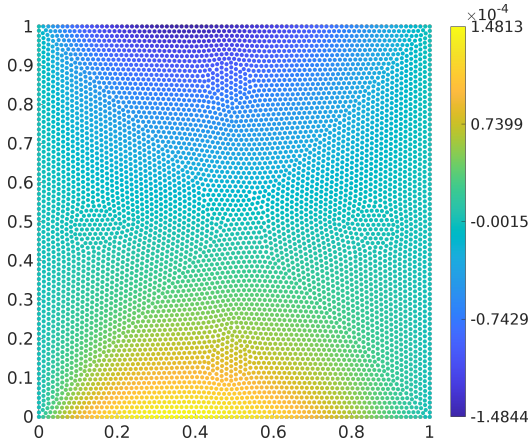
Figure 3.4 plots the deferred correction solution errors in the $||\cdot||_1$ and $||\cdot||_\infty$ norms for the $P2$ approximation. The exact deferred correction computed using the analytic solution is marked by the dotted line. It is observed to converge at the $O(h^4)$ in both norms. For the approximated corrections, all third and fourth order partial derivatives are estimated using the same $P4$ stencil with $n = 40$ neighbors. Using a single deferred correction, we see the errors are significantly lower than the uncorrected $P2$ approximation; however, evidently, there are inaccuracies in constructing estimates of higher order partial derivatives from the

lower order solution. This is clearly seen by jaggedness of the convergence curve bouncing between $O(h^3)$ and $O(h^4)$ lines. To mitigate this behavior, we can use the corrected solution to more accurately approximate the partial derivatives in τ^h. As we see, applying a second deferred correction, the $P2$ stencil - with only $n = 13$ neighbors - converges at a rate close to $O(h^4)$ in the $||\cdot||_1$ norm.

On a final note, in the process of obtaining the correction we also obtain an estimate of the error. In Figure 3.5, we compare the exact pointwise error obtained for the uncorrected $P2$ approximation to the error estimate obtained using Eq. 2.25. As mentioned above, the right hand side vector $\hat{\tau}$ is constructed by estimating the terms given by Eq. 3.6 and Eq. 3.7 using the $P4$ stencil. We observe excellent agreement between the estimated error and the actual error. Here, the maximum absolute errors occur on and near the Neumann boundaries. Figure 3.6 plots the pointwise error for $\hat{\mathbf{e}}^h$. Here, the max differences occurs for points near or on parts of the Neumann boundary that are adjacent to the corners. In these locations, the stencils are heavily asymmetric. Away from the boundaries on the interior, the error in $\hat{\mathbf{e}}^h$ is relatively uniform by comparison.



(a) Estimated pointwise error for the $P2$ approximation by solving Eq. 2.25.



(b) Exact pointwise error for the $P2$ approximation.

Figure 3.5: Error estimation on the meshfree grid corresponding to $N = 80$ nodes per side.

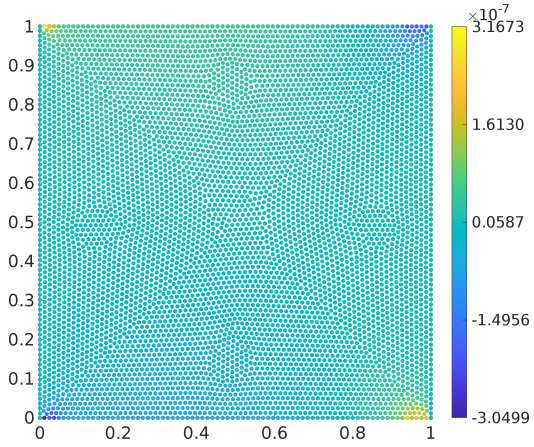
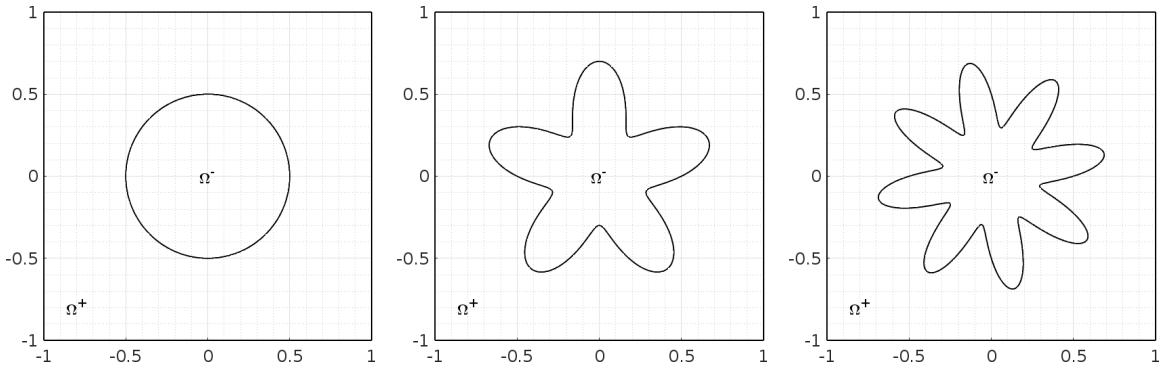


Figure 3.6: Difference between the estimated and exact pointwise error $(\hat{\mathbf{e}}^h - \mathbf{e}^h)$ for Figure 3.5.

3.1.2 Leaflet domain



(a) $\omega = 0$ (b) $\omega = 5$ (c) $\omega = 8$

Figure 3.7: Different domains corresponding to $\omega = 0$, $\omega = 5$ and $\omega = 8$ in Eq. 3.9 with $r_0 = 0.5$.

One of the advantages of collocated meshfree methods is that boundary conditions on complex geometries may easily be handled. In fact, implementation-wise, the following benchmark is not anymore challenging than the preceding benchmarks on the unit square. We consider several irregular domains constructed using the following parametrization [93]

$$\mathbf{x}(\theta) = (r(\theta)cos(\theta))\,\hat{\mathbf{i}} + (r(\theta)sin(\theta))\,\hat{\mathbf{j}}, \quad 0 \leq \theta \leq 2\pi \tag{3.9}$$

where $r(\theta) = r_0 + 0.2sin\,(\omega\theta)$ and r_0 corresponds to the radius of the circle described by the case $\omega = 0$. The interface corresponding to different choices of the parameter ω is shown in Figure 3.7. Here, we see ω sets the number of leaflets. The following manufactured solutions are imposed on respective domains [93]

$$\begin{aligned} T(x,y) &= \frac{r^2}{\beta^-}, \quad (x,y) \in \Omega^- \\ T(x,y) &= \frac{r^4 + C_0 \log(2r)}{\beta^+}, \quad (x,y) \in \Omega^+. \end{aligned} \tag{3.10}$$

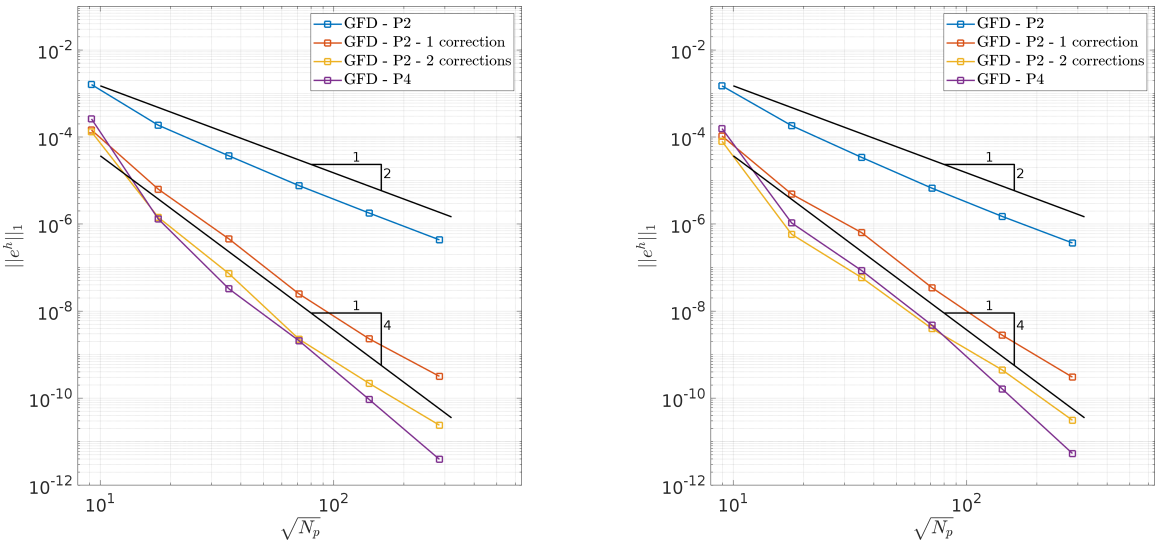
where r is the distance from the origin and where the respective source terms are

$$\begin{aligned} q(x,y) &= \frac{4}{\beta^-}, \quad (x,y) \in \Omega^- \\ q(x,y) &= \frac{16r^2}{\beta^+}, \quad (x,y) \in \Omega^+. \end{aligned} \tag{3.11}$$

With $P2$ shape functions, note the interior problem on Ω^- will be solved exactly to machine precision. As such we focus on solving the exterior problem Ω^+ with Dirichlet conditions on all boundaries. The other parameters used are: $r_0 = 0.5$, $C_0 = -0.1$ and $\beta^+ = 10$. We consider meshfree grids with a total number of nodes ranging from $N_p = 80$ to $N_p = 80,700$ nodes. These quasi-uniform meshfree grids are generated using the approach discussed in Section 2.6, with the leaflet boundary represented as an implicit function.

The convergence rates in $||\cdot||_1$ norm are shown in Figure 3.8 for the $P2$ approximation $(n = 13)$ with and without deferred corrections and are compared to the $P4$ approximation $(n = 40)$. For the deferred corrections, the higher order partial derivatives are estimated using the the $P4$ shape functions. The Dirichlet boundary conditions are imposed exactly so no corrections are necessary for the boundaries. As we can see, the $P2$ approximation gives second order accurate behavior. This convergence rate is again inline with the $||\cdot||_1$ norm of the leading truncation error term. Applying one correction, the error significantly drops with a fourth order convergence rate for a wide range of resolutions until it tapers off on the finest resolution. In fact, for $\omega = 5$, the $P2$ approximation with one deferred correction

on the grid comprised of $N_p = 1260$ grid points is more accurate than the uncorrected $P2$ approximation constructed on the $N_p = 80700$ node grid. Applying a second correction, the error again drops but eventually exhibits the same tapering behavior for fine resolutions.

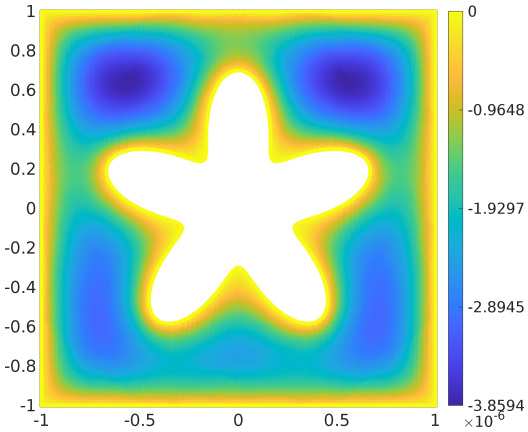


(a) Convergence rates for case $\omega = 5$.

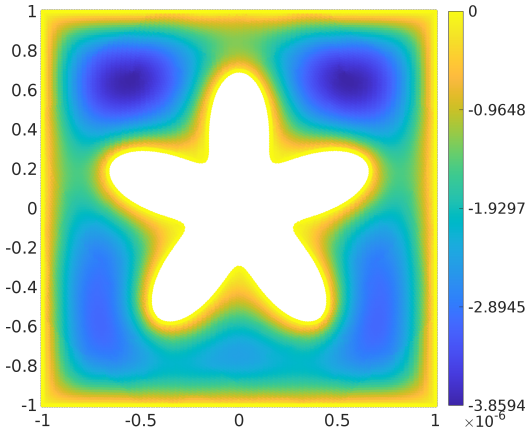
(b) Convergence rates for case $\omega = 8$.

Figure 3.8: Convergence rates of the average absolute solution error $||\mathrm{e}^h||_1$ for $\omega = 5$ (left) and $\omega = 8$ (right) using the $P2$ approximation with and without deferred corrections compared to the $P4$ approximation.

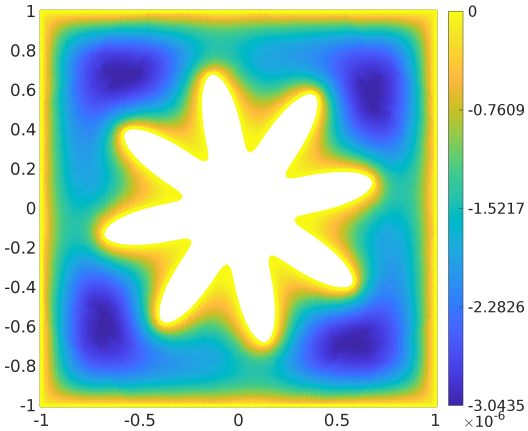
In Figure 3.9 the pointwise error estimate $\hat{\mathrm{e}}^h$ is plotted for the $\omega = 5$ and $\omega = 8$ cases and compared to the exact pointwise error for the grid with approximately 20,000 nodes. Here, the absolute error is smallest near the Dirichlet boundaries and is exactly zero on the boundaries themselves. As before, we still find excellent agreement between the exact pointwise error and estimated errors. Indeed, from Figure 3.10, we can conclude that the max relative error for the error estimate is on the order of 1×10^{-3} for both cases.



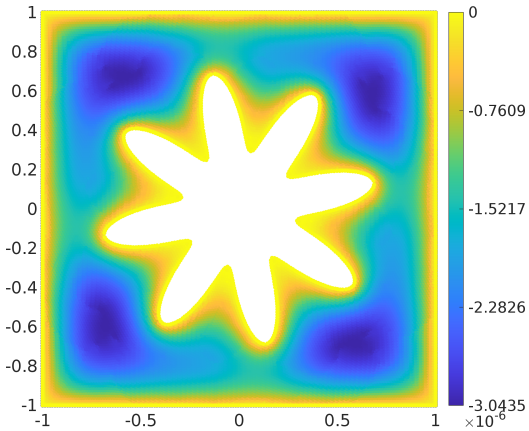
(a) Leaflet $\omega = 5$, estimated pointwise error for the $P2$ approximation (Eq. 2.25).



(b) Leaflet $\omega = 5$, exact pointwise error for the $P2$ approximation.

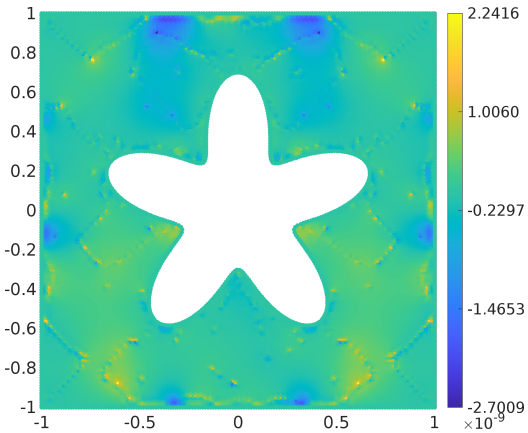


(c) Leaflet $\omega = 8$, estimated pointwise error for the $P2$ approximation (Eq. 2.25).



(d) Leaflet $\omega = 8$, exact pointwise error for the $P2$ approximation.

Figure 3.9: Error estimation on the meshfree grid corresponding to approximately 20,000 grid nodes.



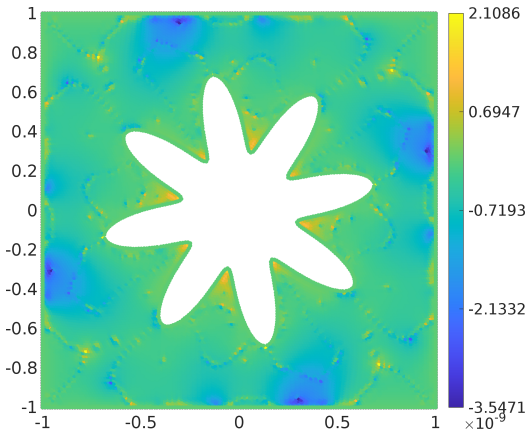


Figure 3.10: Difference between the estimated and exact pointwise error $(\hat{\mathbf{e}}^h - \mathbf{e}^h)$ for leaflets $\omega = 5$ (left) and $\omega = 8$ (right) in Figure 3.9.

3.1.3 L-shape domain

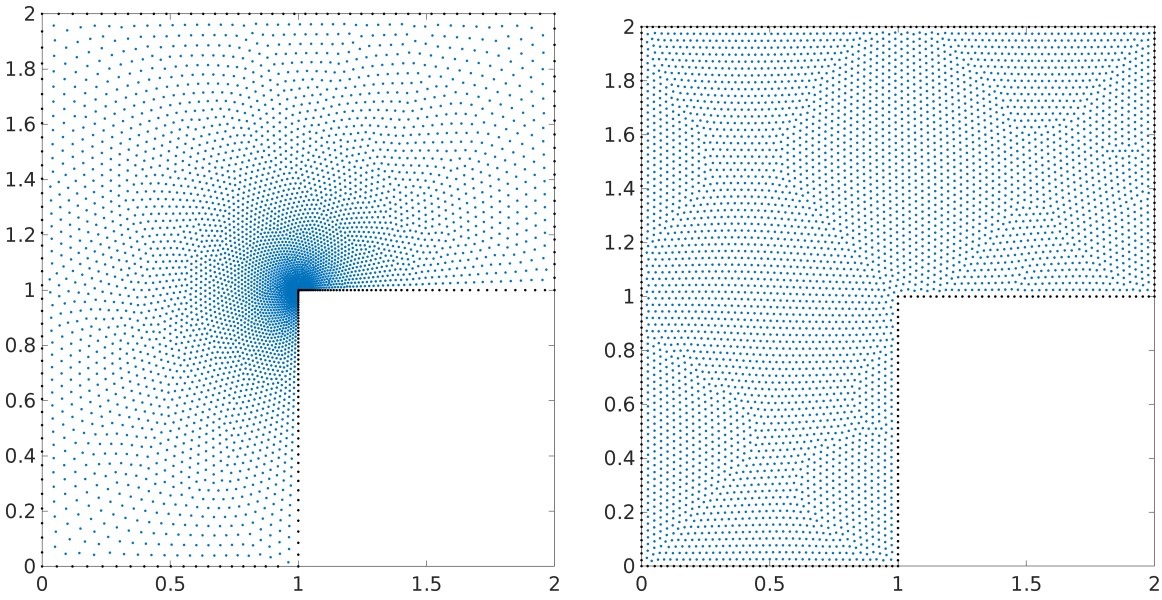


Figure 3.11: Example adaptive (a) and uniform (b) meshfree grids for the Laplace problem on the L-shape domain. For this adaptive grid the cluster ratio is $\Delta x_{max}/\Delta x_{min} = 8$ with 4417 grid points, whereas the uniform grid has 4806 points.

We consider Laplace's equation $\nabla^2 u = 0$ on the L-shape domain shown in Figure 3.11. Dirichlet conditions are imposed according to the analytic solution, which in polar coordinates is [50]

$$u(r, \theta) = r^{2/3} sin(2\theta/3). \tag{3.12}$$

Here, r is the radius as measured from the non-convex corner. Due to the discontinuous derivatives at the corner, the regularity of the solution is reduced and as a result the spatial convergence order will be reduced. To overcome the reduced convergence, we consider geometric adaptivity about the singularity using the signed distance function $d_a(\mathbf{x}) = -||\mathbf{x} - \mathbf{x}_c||$, where $\mathbf{x}_c$ is the location of the non-convex corner. An example adaptive grid is shown in Figure 3.11a using a clustering ratio of $\Delta x_{max}/\Delta x_{min} = 8$. Six control points are introduced at the corners such that the following approximate signed

distance function can be used to represent the domain

$$d_\Gamma(\mathbf{x}) = \max\left(d_1(\mathbf{x}), -d_2(\mathbf{x})\right), \tag{3.13}$$

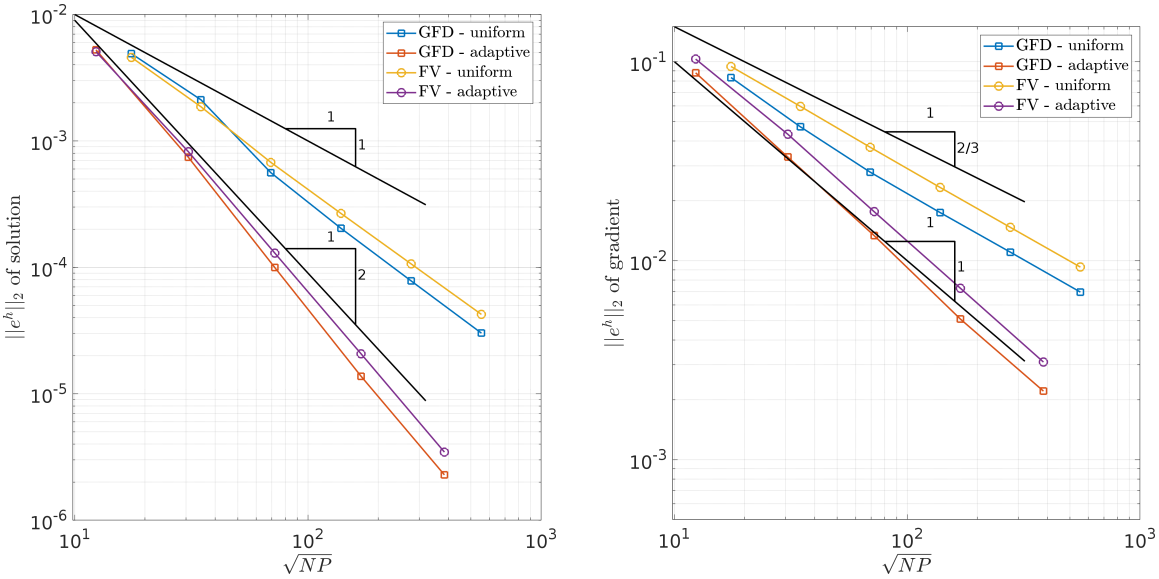
where $d_1(\mathbf{x})$ is the signed distance function to the outer square and $d_2(\mathbf{x})$ is the signed distance function of the inner square.

We will compare the GFD solution using the $P2$ shape functions on uniform and adaptive grids to a second order median dual finite volume (FV) [94] solution obtained on the same grids. Specifically, for the FV solution, we construct a Delanauy triangulation and then obtain a local flux conservation statement over each median dual control volume in the domain (i.e., one for each vertex). The solution is assumed piecewise linear in the primary mesh and so each triangular element contributes a constant flux to the two faces of the corresponding median-dual control volume.

The $||\cdot||_2$ norm of the error is compared to the median dual FV errors in Figure 3.12. For GFD, the $||\cdot||_2$ norms were computed as defined in Eq. 2.57 using degree four quadrature, see Section 2.3. The $||\cdot||_2$ norm of errors for the FV solutions were also evaluated in a similar manner; however, no reconstruction was necessary, since unlike GFD, the median dual FV solution is global.

For the quasi-uniform grids, we observe the FV convergence orders are reduced to $O(h^{1.36})$ and $O(h^{0.67})$ for the solution and gradient, respectively. Note that for steady diffusion, this median dual finite volume approach is equivalent to linear finite elements (FE) [94]. As such, we see the gradient error of the quasi-uniform grid median FV solutions agrees with the theoretical limit $O(h^{2/3})$ [50]. The quasi-uniform grid GFD solution scales similarly with $O(h^{1.5})$ and $O(h^{0.7})$, respectively. Surprisingly, the GFD solution is slightly more accurate than the median dual FV (linear FE) solutions despite its non-conservative nature. Note we have chosen the x-axis to be the square root of the total number of grid points. With respect to this x-axis, an adaptive scheme is effectively second order (in 2D)

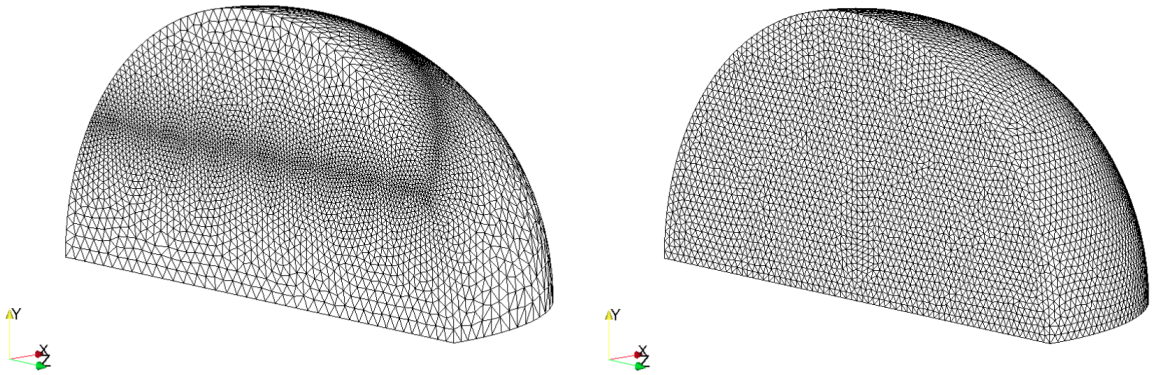
if quadrupling the total number of points decreases the error by a factor of four. Both the adaptive GFD and FV solutions recover second order convergence rates for the solution error and first order convergence rates for the gradient error. For this problem, the adaptive GFD solution is more accurate than the median dual FV solution for the solution as well as the gradient while using the exact same grids. However, the GFD stencil uses almost twice the number of neighbors as the linear FV stencils.



(a) (b)

Figure 3.12: The $||\cdot||_2$ error for the solution (a) and the gradient (b) compared to the median dual FV solutions (linear) using the same quasi-uniform and adaptive grids. The x-axis is the square root of the total number of points. The adaptive grids correspond to clustering ratios of $\Delta x_{max}/x_{min} = 2, 4, 8, 16$ and 32 in Eq. 2.95 with $\Delta x_{min} \approx 0.1, 2.5\times10^{-2}, 6.25\times 10^{-3}, 1.56 \times 10^{-4}$ and 3.91×10^{-5}. GFD computations used with $P2$ approximation and $n = 13$ nearest neighbors.

3.1.4 Quarter sphere domain



(a) Adaptive meshfree grid.

(b) Quasi-uniform meshfree grid.

Figure 3.13: Example adaptive (a) and quasi-uniform (b) meshfree grids for the hyperbolic tangent problem on the quarter sphere domain with radius $R = 0.5$. For the adaptive grid $\Delta x_{max}/\Delta x_{min} = 16$ with 50158 grid points and $\Delta x_{min} = 1/160$, whereas the uniform grid has $\Delta x_{min} = 1/70$ with 53348 points. Note the grids are meshfree and a tetrahedral mesh has been added to aid the eye.

We consider the Poisson equation on a quarter of a sphere (unit diameter) as shown in Figure 3.13. These 3D meshfree grids are also generated using the grid generation approach discussed in Section 2.6. We consider a manufactured a solution such that source term and Dirichlet conditions satisfy the hyperbolic tangent function

$$u\left(\mathbf{x}\right) = \tanh\left(-\alpha\left(\mathbf{x} - \mathbf{x}_p\right) \cdot \hat{\mathbf{n}}\right), \tag{3.14}$$

where $\hat{n}$ is the unit normal of the plane going through the point $\mathbf{x}_p$, and α is a scaling term that determines the length scale $w \approx 2/\alpha$ over which the hyperbolic tangent function varies from -1 to 1. For this unit diameter wedge, we set $\alpha = 40$ such that most of the variation in the solution occurs over a width about a twentieth of the diameter of the sphere $w \approx \dfrac{1}{20}$. We choose the variation to occur normal to the plane defined by $\mathbf{x}_p = (0, 0.25, 0.5)$ with unit normal $\hat{\mathbf{n}} = (-1/\sqrt{2}, 1/\sqrt{2}, 0)$. Correspondingly, the equilibrium length in Eq. 2.95 is defined by the signed distance function to the plane $d_a\left(\mathbf{x}\right) = \min(d_1, -d_1)$, where $d_1\left(\mathbf{x}\right) = \left(\mathbf{x} - \mathbf{x}_p\right) \cdot \hat{n}$. As can be seen in Figure 3.13, the meshfree grid generation approach correctly

spreads the points on the surface for both the quasi-uniform and non-uniform cases. On the interior, the boundary domain is represented using the following approximate signed distance function

$$d_\Gamma(\mathbf{x}) = \max(\max(d_1(\mathbf{x}), d_2(\mathbf{x})), d_3(\mathbf{x})) \tag{3.15}$$

where $d_1(\mathbf{x})$ is the signed distance function for the sphere and where $d_2(\mathbf{x})$ and $d_3(\mathbf{x})$ are the signed distance functions to the planes normal to the x-axis and y-axis, which quarter the unit sphere. On the exterior, the exact distance functions are used to ensure points are projected back onto the boundary at the intersection of surfaces.

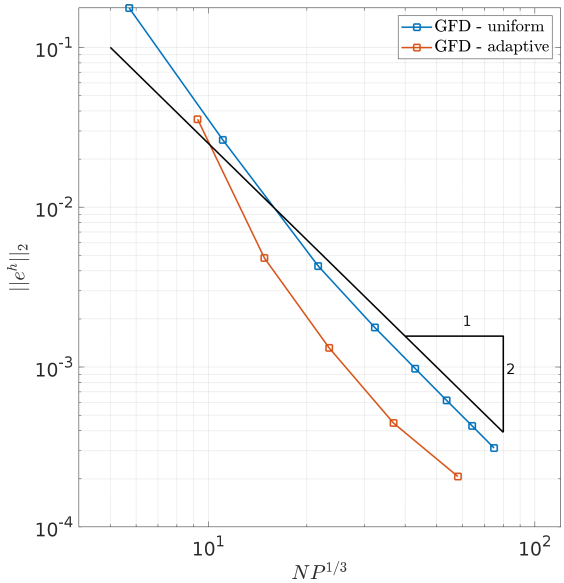


Figure 3.14: L_2 solution error for the hyperbolic tangent problem on a quarter sphere using GFD on uniform and adaptive grids. Adaptive grids correspond to clustering ratios $\Delta x_{max}/\Delta x_{min} = 2, 4, 8, 16$ and 32 with $\Delta x_{min} = 1/20, 1/40, 1/80, 1/160$ and $1/320$ resulting in 793, 3261, 12823, 50158 and 196,054 points respectively.

Figure 3.14 compares the adaptive and uniform solution errors in the $||\cdot||_2$ norm. Here, the $n = 19$ nearest neighbors were used with the $P2$ approximation. The L_2 error norm $||\cdot||_2$, as given by Eq. 2.56 was evaluated using degree 3 Gauss quadrature on a tetrahedralization of the meshfree points as discussed in Section 2.3. As we can see, for this manufactured problem the adaptive solution is more accurate than the uniform solution. For instance, the adaptive solution corresponding to $\Delta x_{min} = 1/160$ has 50,158 points and

results in a solution error of 2.06×10^{-4}, whereas the uniform grid with $\Delta x_{min} = 1/120$ results in 265,630 points and twice the error (4.28×10^{-4}). However, whereas the uniform grid maintains second order convergence throughout, the adaptive grid's convergence rate is observed to be sensitive to the grid clustering ratio. For the largest grid clustering ratios of $\Delta x_{max}/\Delta x_{min} \approx 16$ and 32, we observe that the slope tapers off from a rate greater than second order to approximately $O(h^{1.7})$.

3.2 Parabolic equation

In this section, we will apply GFD to solve the transient diffusion equation in 2D,

$$\frac{\partial T}{\partial t} = \nabla \cdot k \nabla T + q, \tag{3.16}$$

where the time derivative is evaluated at a fixed spatial coordinate and we again fix $k = 1$. We will use this scalar diffusion equation to 1) verify the implicit BDF time integration schemes as they will be re-used for time marching the momentum equations in the incompressible flow simulations and 2) illustrate two different approaches to measuring error for time dependent problems.

3.2.1 Square domain - revisited

We revisit the simple 2D unit square domain with the same quasi-uniform grids used previously. A manufactured solution is constructed such that the initial conditions, boundary conditions, and source term satisfy the analytic solution

$$T(\mathbf{x}, t) = \left(1 + e^{-t}\right) T_{ss}(\mathbf{x}) \tag{3.17}$$

where T_{ss} is the steady state solution, here chosen to coincide with Eq. 3.2. The same set of boundary conditions are chosen as before, namely Dirichlet conditions on the left and right and Neumann conditions on the top and bottom.

For time integration, we consider three implicit BDF-k schemes [84], where $k = 1, 2$ and $k = 3$ and indicates the temporal convergence order as well as the polynomial degree. For a given node i, the BDF-k schemes correspond to constructing a degree k polynomial about the time point t^{n+1} that interpolates the values at T_i^{n+1-q} where $q \in [0, k]$. This polynomial is then differentiated with respect to time and evaluated at $n + 1$

$$\frac{\partial T_i}{\partial t} = \sum_{q=0}^{k} \left.\frac{d\psi_q}{dt}\right|_{t^{n+1}} T_i^{n+1-q}, \tag{3.18}$$

where $\alpha_q = \left.\dfrac{d\psi_q}{dt}\right|_{t^{n+1}}$ is the qth polynomial shape function differentiated and evaluated at t^{n+1}. For uniform time stepping, the BDF coefficients α_q are summarized below

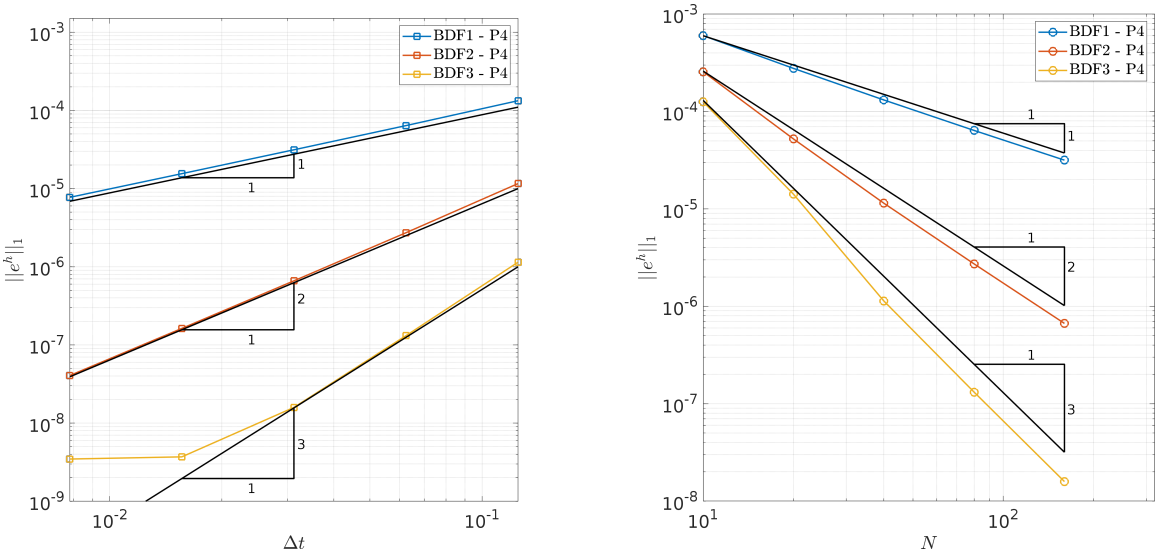
$$\text{BDF-1} : \left\{ \alpha_0 = \frac{1}{\Delta t}, \alpha_1 = -\frac{1}{\Delta t} \right. \tag{3.19}$$

$$\text{BDF-2} : \left\{ \alpha_0 = \frac{3}{2\Delta t}, \alpha_1 = -\frac{2}{\Delta t}, \alpha_2 = \frac{1}{2\Delta t} \right. \tag{3.20}$$

$$\text{BDF-3} : \left\{ \alpha_0 = \frac{11}{6\Delta t}, \alpha_1 = -\frac{3}{\Delta t}, \alpha_2 = \frac{3}{2\Delta t}, \alpha_3 = -\frac{1}{3\Delta t} \right. \tag{3.21}$$

When starting off the integration for the higher order BDF schemes, the appropriate lower order BDF schemes can be first used until sufficient previous values are available. For instance, the BDF-3 will first require using BDF-1, then at the next time step the BDF-2 scheme, after which the BDF-3 scheme can be used. To measure the error we use the $||\cdot||_1$ norm in both space and time under the time interval $t \in [1.5, 5]$. Two different approaches are considered. The first approach show in Figure 3.15a is the standard approach. In this approach, a sufficiently small Δx is chosen and fixed while the time step is varied. Specifically, we use $\Delta x \approx \dfrac{1}{80}$ while Δt is halved starting from $\Delta t = 0.125$. Note that we have avoided a very fine spatial resolution by choosing to use the $P4$ WLS shape functions for the spatial discretization, which result in much lower errors than the $P2$ approximation. As we can see, all three schemes converge at the expected order. For small enough time steps, the third order BDF-3 scheme starts stagnating as the temporal truncation errors

become lower than the $P4$ spatial truncation errors. Hence the error being measured is dominated by the spatial truncation errors which do not decrease as we have fixed Δx. This crossing point would occur much earlier if the $P2$ approximation were used or if we tried to measure the convergence rate of the BDF-4 scheme.



(a) Solution error in $||\cdot||_1$ norm with fixed spatial resolution $\Delta x \approx 1/80$ while halving the time step Δt starting with $\Delta t = 0.125$.

(b) Solution error in $||\cdot||_1$ norm with time step and spatial resolution both halved such that $\Delta t \propto \Delta x$.

Figure 3.15: BDF temporal convergence rates in the $||\cdot||_1$ norm for BDF-1, BDF-2 and BDF-3 while using the $P4$ shape functions for $t \in [1.5, 5]$.

As shown in Figure 3.15b, an alternative approach, is to instead vary the time step with the spatial resolution. Specifically, here we use $\Delta t \propto \Delta x$. As the temporal schemes are lower order than the spatial scheme used here, with refined resolutions, the dominant error measured will be the temporal order. This approach can also be used to verify equal order time and space discretizations but no definitive conclusion can be drawn about which error is measured. In this case, it can only be verified that the error as a whole is converging at the expected rates. This is a useful approach, as you can simultaneously verify the spatial and temporal convergence orders of equal order schemes.

With regards to stability, in all these simulations we are stable while time marching with $\Delta t \propto \Delta x$. Consequently, we are not restricted by the diffusive time step $\Delta t \propto \Delta x^2$ of the

explicit Euler scheme. Indeed, the time step can be kept fixed but this is not recommended as the temporal errors will quickly dominate the solution.

3.3 Incompressible flows

In this section, two steady incompressible flow benchmarks are validated on fixed, clustered meshfree grids. Namely, the lid driven cavity and uniform flow over a cylinder. The incompressible Navier-Stokes equations are marched towards steady state using an approximate projection method. The boundary conditions are implemented using a slightly different approach than discussed here in this thesis and are based on ghost nodes as is detailed in our previous publications [6, 95].

3.3.1 Classic lid driven cavity

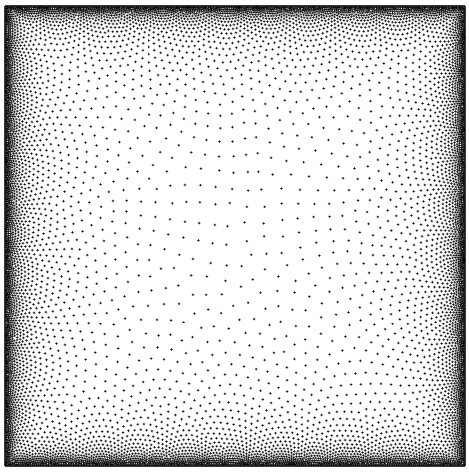
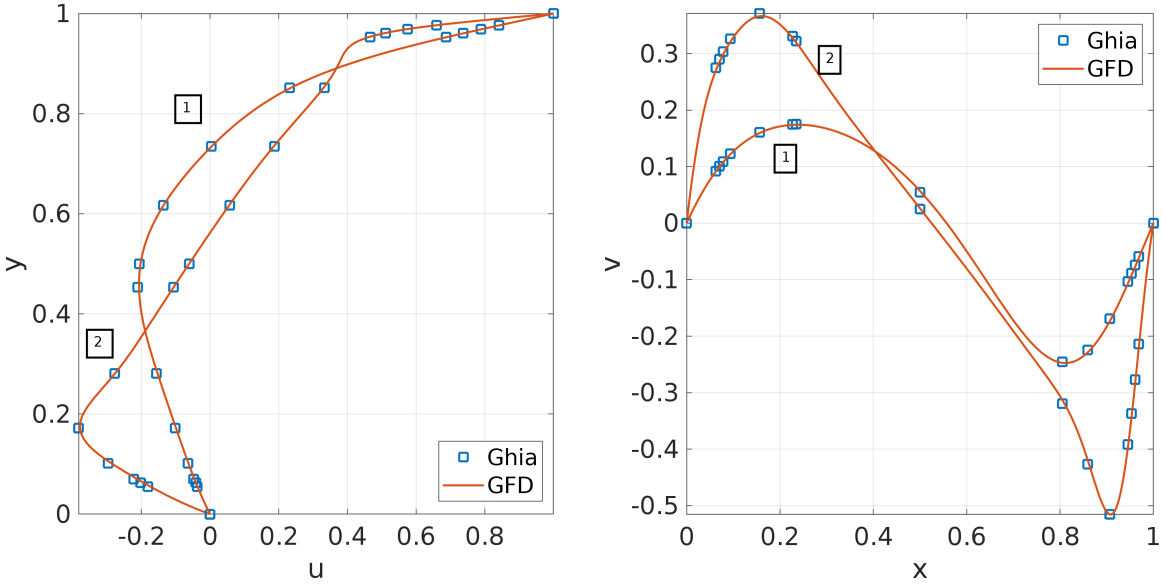


Figure 3.16: Meshfree grid used for the classic lid driven cavity corresponding to $Re = 100$. Clustering ratio of approximately 12 to 1.

We consider the classic lid driven cavity problem involving an incompressible flow in a square cavity with a constant lid speed and other walls fixed [96]. Below a critical Reynolds number the flow is steady. To resolve the viscous boundary layer we use variable resolution meshfree grids. The grid clustering is achieved based on distance functions to the four

boundaries. Figure 3.16 shows the corresponding grid for $Re = 100$. The variable resolutions used here are particular useful in resolving the velocity discontinuities in the upper two corners. The equations are marched to steady state using a time step Δt such that a Courant number of $C_o = 0.5$ is satisfied on the smallest spacing.

In Figure 3.17, the steady GFD solutions are compared to the Ghia solutions for $Re = 100$ and $Re = 1000$. As shown, the velocity profiles along the vertical and horizontal centerlines agree well with the Ghia data set for both Reynolds numbers. The meshfree grid corresponding to $Re = 100$ has a clustering ratio of 12 to 1 with approximately 11,000 total fluid nodes, while for $Re = 1000$ the grid has a clustering ratio of 32 to 1 with approximately 43,000 fluid nodes.



(a) Horizontal velocity component along vertical centerline.

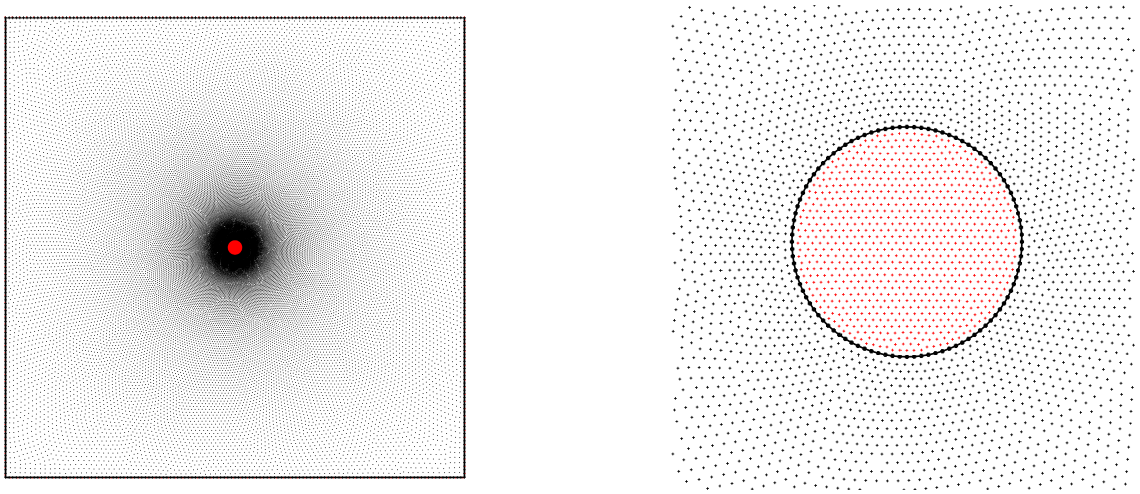
(b) Vertical velocity component along horizontal centerline.

Figure 3.17: GFD sampled solution compared to Ghia solutions for the lid driven cavity. The labels 1 and 2 correspond to $Re = 100$ and $Re = 1000$, respectively.

3.3.2 Flow over a cylinder

The configuration tested is steady 2D uniform flow over a cylinder in an "unbounded" domain with the Reynolds number defined as $Re = U_\infty D/\nu$, where D is the cylinder diameter, U_∞ the upstream velocity and ν the kinematic viscosity. The test case has been

thoroughly investigated by others and is frequently used to validate different numerical approaches. It is generally agreed that past a critical Reynolds number of $Re_c \approx 46$ the two standing symmetric vortices detach from the cylinder surface and begin to oscillate [97]. We consider the steady case for $Re = 5, 10, 20$ and 40.



(a) Zoomed out.

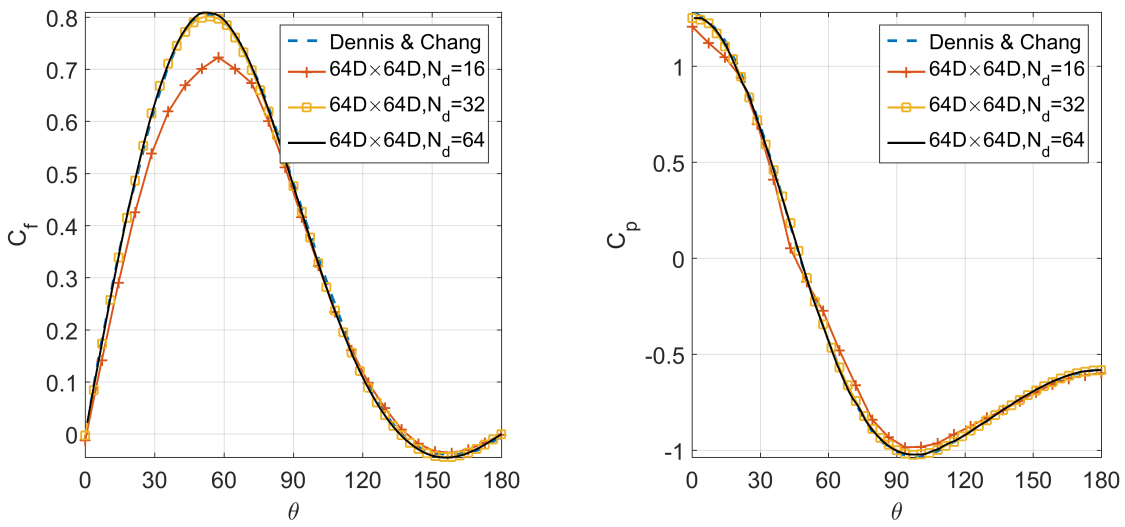
(b) Near the cylinder surface.

Figure 3.18: Grid corresponding to a domain size of $32D \times 32D$ with a clustering ratio of 16 with 32 nodes across the diameter. Nodes on the interior of the circle are treated as the ghost nodes for sharp interface boundary conditions.

To consider boundary effects we simulated several different sized square domains ($16D \times 16D$, $32D \times 32D$, and $64D \times 64D$) with the cylinder positioned in the center. As the cylinder dimensions are small relative to the computational domain, it is necessary to consider a variable resolution grid as shown in Figure 3.18 in order to resolve the viscous flow around the cylinder. At boundaries where the velocity is specified we set $\partial p/\partial n = 0$. This condition is enforced at the inlet ($u = U_\infty = 1$), cylinder surface ($u, v = 0$), and top and bottom of the square domain ($u = U_\infty, v = 0$). At the outlet, we assume Neumann conditions on both components of velocity ($\partial u/\partial n = \partial v/\partial n = 0$) and set $p = 0$.

We determined the dimensions of the domain that mimic an unbounded domain by varying the domain size from $16D \times 16D$ to $64D \times 64D$ while keeping a fixed resolution near the cylinder surface. We found the velocity gradients near the cylinder surface changed

minimally for domains greater than $64D \times 64D$.



(a) Skin friction coefficient.

(b) Pressure coefficient.

Figure 3.19: Skin friction coefficient C_f and pressure coefficient C_p at the cylinder surface for $Re = 20$ as the resolution increases on the $64D \times 64D$ domain. GFD results compared to Dennis and Chang's boundary fitted solution [98].

Figure 3.19 shows we converge to the Dennis and Change solution [98] as the resolution near the cylinder surface is doubled twice from $N_d = 16$ nodes across the diameter to $N_d = 64$ nodes with a fixed clustering ratio of 32:1. The skin friction coefficient curve, which involves computing derivatives, is smooth with no jumps present. The grid clustering ratio of 32:1 decreased the computational costs significantly by decreasing the number of nodes by two orders of magnitude from approximately 1.6×10^7 nodes on the uniform grid to approximately 1.6×10^5 nodes on the clustered grid. We were unable to reliably use higher clustering ratios due to instabilities near the inlet. We suspect these instabilities are rooted in the simple treatment of the convective term which is analogous to centered differences on a regular grid. The same convergence behavior is observed for $Re = 5, 10$ and 40.

Based on the results for C_f and C_p with $Re = 20$, we decided a grid size of $64D \times 64D$ was sufficient to use for our purposes. In Table 3.1, we compare the GFD results to Dennis

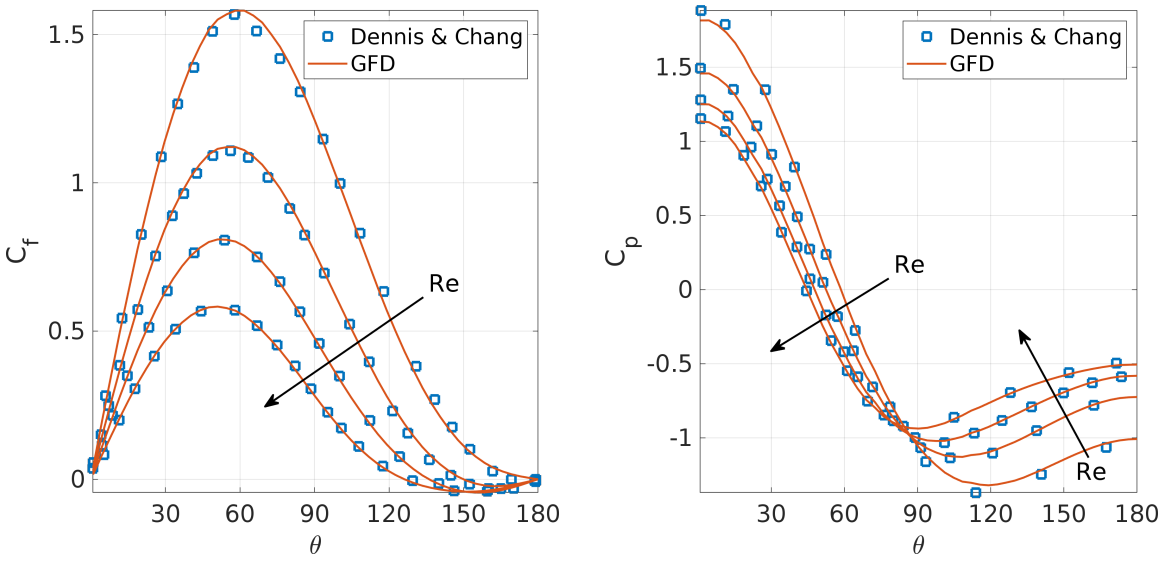
and Chang's boundary fitted results using the relative error. We see GFD solution matches most of the parameters to within 1×10^{-2} relative error throughout the range of Reynolds number considered with the most disagreement occurring for the recirculation lengths and stagnation pressures. We did not observe separation for $Re = 5$, agreeing with Dennis and Chang's prediction that separation occurs for a Reynolds number greater than $Re_c = 6.2$.

Table 3.1: GFD flow parameters (bold) with relative errors (below) calculated with respect to Dennis and Chang's boundary fitted results [98]. A $64D \times 64D$ domain with a clustering ratio of 32:1 and $N_d = 64$ nodes across the diameter was used.

Re	L_r/D	θ_s	F_d	P_d	C_d	$p(0)$	$p(\pi)$
5	—	—	**1.932**	**2.145**	**4.077**	**1.815**	**-1.009**
			-7.76E-3	2.52E-2	9.57E-3	3.14E-2	3.47E-2
10	**0.247**	**149.9**	**1.251**	**1.580**	**2.831**	**1.458**	**-0.725**
	7.20E-2	3.34E-3	-4.00E-3	1.27E-2	5.30E-3	2.13E-2	2.36E-2
20	**0.919**	**136.1**	**0.815**	**1.229**	**2.044**	**1.251**	**-0.582**
	2.34E-2	1.47E-3	-3.19E-3	-3.25E-3	4.89E-4	1.44E-2	1.24E-2
40	**2.259**	**126.2**	**0.525**	**1.002**	**1.528**	**1.135**	**-0.506**
	4.03E-2	0.0	-1.90E-3	-3.99E-3	-3.93E-3	7.93E-3	5.33E-3

Figure 3.20 shows GFD successfully estimates the pressure coefficient C_p and skin friction coefficient C_f along the cylinder surface despite the irregular grid and despite using only the 13 nearest neighbors. The meshfree grid corresponds to a domain size of $64D \times 64D$, $N_d = 64$ nodes across the diameter, and a clustering ratio of 32:1.

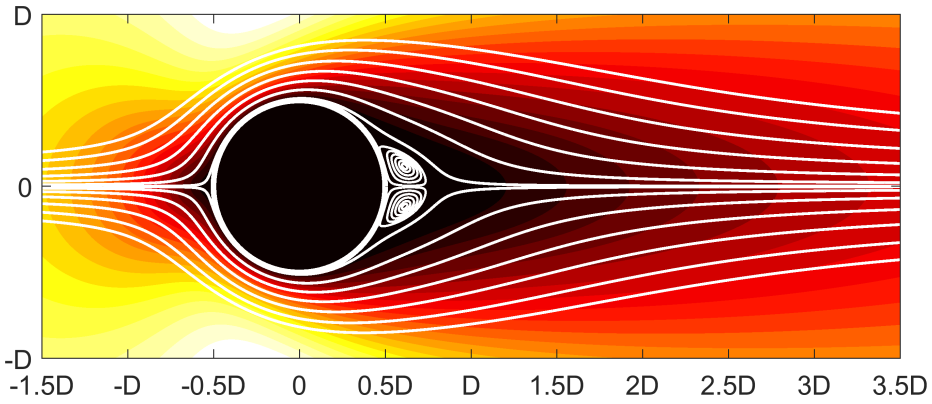
As shown in Figure 3.21, we may also recover streamlines and velocity magnitude contours after sampling the numerical solution onto a regular grid. Here, the numerical solution from the irregular grid was linearly sampled onto a 321×129 lattice patch sized approximately $5D \times 2D$ using linear WLS shape functions with a small compact support of 13 neighbors.



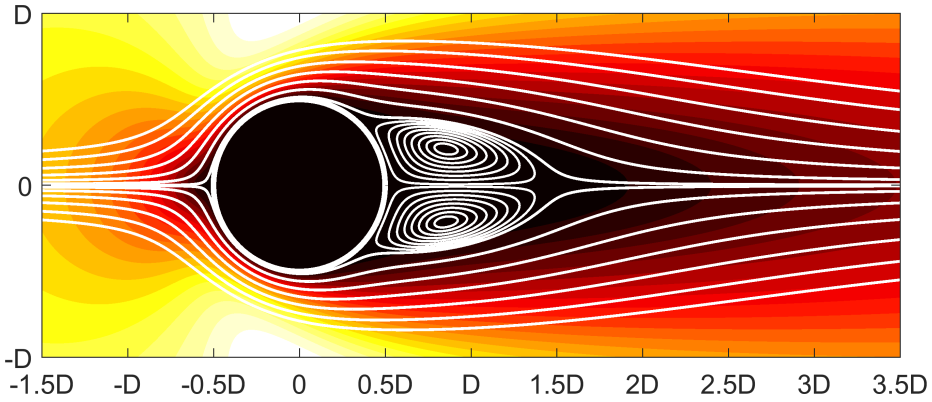
(a) Skin friction coefficient.

(b) Pressure coefficient.

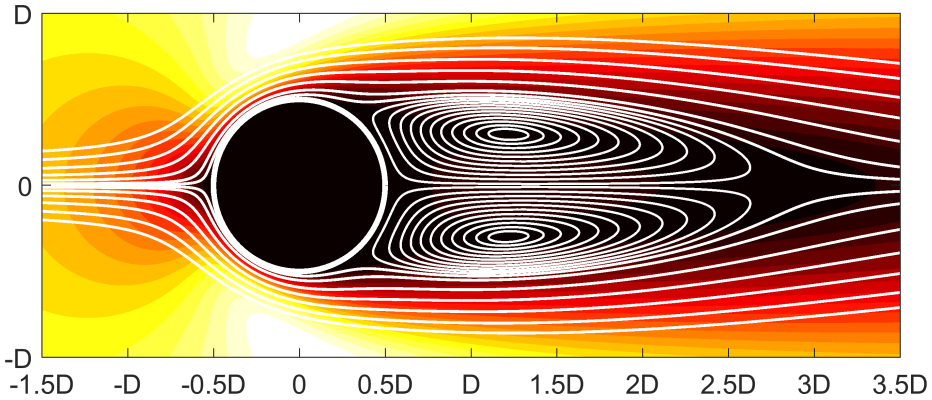
Figure 3.20: Skin friction coefficient C_f and pressure coefficient C_p at the cylinder surface for $Re = 5, 10, 20, 40$ compared to Dennis and Chang's boundary fitted solution [98].



(a) Re = 10



(b) Re = 20



(c) Re = 40

Figure 3.21: GFD solution streamlines and velocity magnitude contours.

CHAPTER 4

MOVING QUASI-UNIFORM MESHFREE GRIDS

In this section, moving quasi-uniform meshfree grids are used to solve several incompressible flow problems. Namely, Taylor-Green vortex decay, the modified lid driven cavity, and the inline-oscillating cylinder. The fixed meshfree grids are rezoned and remapped based on the approach mentioned in Section 2.5.2 and as detailed in the publication [6]. The first two cases have analytic solutions and are used to verify the method for both fixed and moving grids. On the other hand, the inline oscillating cylinder does not have an analytic solution. To validate, a comparison to the lattice Boltzmann method and other published numerical results is made.

4.1 Taylor-Green vortex decay

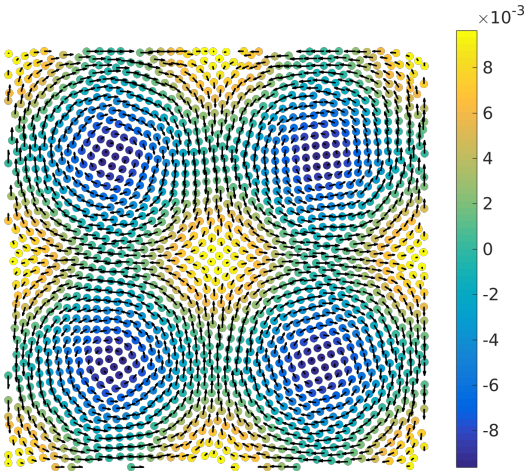


Figure 4.1: Taylor-Green vortex moving meshfree grid solution for $N \approx 40$ with grid regularization at a decay time $t^* = 2$. Here, $N \approx \sqrt{N_p}$. Velocity vectors overlaid with nodes that are colored according to the pressure value.

The Taylor-Green vortex problem consists of simulating the decay of a periodic array of vortices initialized at $t = 0$. The non-dimensional solution for a period length of $L = 1$ is

given by

$$
\begin{aligned}
u &= sin(2\pi x)cos(2\pi y)f(t), \\
v &= -cos(2\pi x)sin(2\pi y)f(t), \\
p &= 0.25f(t)^2\left(cos(4\pi x) + cos(4\pi y)\right),
\end{aligned}
\tag{4.1}
$$

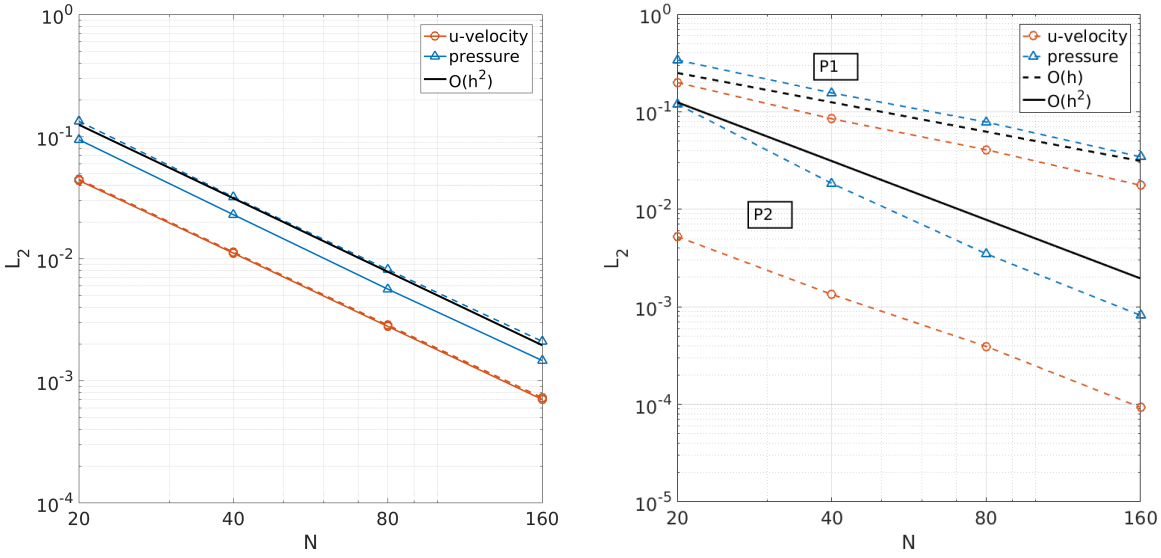
where $f(t) = exp(-8\pi^2 t/Re)$ with $Re = UL/\nu$ the Reynolds number. We impose periodic boundary conditions on velocity and pressure in both spatial directions.

Figure 4.2 plots the time averaged L_2 relative error for the BDF-1 and BDF-2 approximate projection scheme. Using a direct approach [99], Figure 4.2a, compares the fixed and moving grid errors for the BDF-1 scheme while simultaneously changing the grid spacing and time step. Note this approach to measuring error was shown earlier for the transient heat equation. We expect the BDF-1 scheme with second order GFD spatial discretization to be second order in space and first order in time. If a convective time scale ($\Delta t \propto \Delta x$) is used, we expect first order errors will be measured; however, with the direct approach only the lowest order of the scheme will be measured (spatial or temporal). To determine that the spatial errors are not first order, the diffusive time scale ($\Delta t \propto \Delta x^2$) can be used to ensure the expected first order temporal and splitting errors of the BDF-1 scheme do not mask the spatial errors. In this manner, the asymptotic convergence behavior can be evaluated.

As can be seen in Figure 4.2a, rezoning and remapping the fixed grid solutions using a linear approximation $P1$ does not change the expected second order spatial rate and first order temporal rate of BDF-1. Comparing the moving meshfree grids to the fixed meshfree grids, we see the former have slightly increased errors on the pressure but maintain the same convergence rate.

In Figure 4.2b, we compare a third order (P2) and second order (P1) remapping for the BDF-2 scheme while using the convective time step ($\Delta t \propto \Delta x$) on moving grids. It is observed that the local errors introduced to the velocity field during the remapping procedure from the original fixed grid $\mathbf{x}_n$ to new updated and shifted grid $\mathbf{x}_s$ must be third

order (P2). A linear update (P1) will reduce BDF-2 to a first order scheme. Figure 4.2b also confirms that the incremental pressure form used in BDF-2 results in a second order splitting error since second order convergence is maintained while using $\Delta t \propto \Delta x$. For either scheme, if the grid regularization is not performed, voids and undesirable clustering develop leading to unstable GFD stencil coefficients. Similar instabilities are reported using SPH [19, 37, 38]. For specific illustrations of Lagrangian grid degradation for the Taylor-Green vortex decay, see Xu et al.[37] or Oger et al. [19]



(a) BDF-1 scheme with $\Delta t \propto \Delta x^2$.

(b) BDF-2 scheme with $\Delta t \propto \Delta x$.

Figure 4.2: Time averaged L_2 relative error norm as a function of the approximate resolution $N \approx \sqrt{N_p}$ for $Re = 10$. Fixed grid (solid), moving grid (dotted).

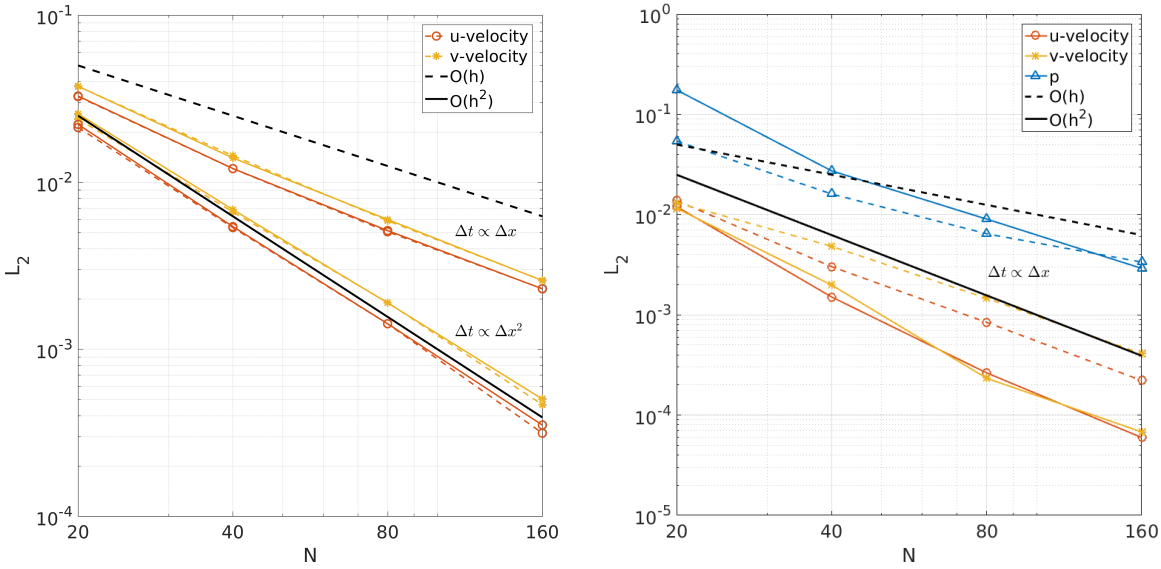
4.2 Modified lid driven cavity

The classic lid driven cavity problem involves an incompressible flow in a square cavity with a constant lid speed and other walls fixed [96]. Near the corners, the constant lid speed leads to a discontinuous velocity field. As a result, properly evaluating the numerical accuracy of a scheme using this benchmark is difficult [100]. For moving meshfree grids, the singularity poses additional difficulties as particles in the top right corner penetrate the boundary [38]. While boundary penetration can be detected and handled accordingly, we

find the singularity results in instabilities for our moving meshfree grid solver even for a small Reynolds number (i.e., $Re = 100$). If instead fixed meshfree grids are considered, instabilities do not arise and the steady state solution may be resolved for a range of Reynolds numbers using grid clustering as was shown in Section 3.3.1.

Alternatively, the lid driven cavity may be regularized by prescribing a continuous lid speed. We consider two cases of the modified problem. In Case 1, the method of manufactured solutions [101] is used to construct an analytic solution. The steady state solution considered here is detailed by Shih and Tan [102] and has velocity components

$$\begin{aligned} u(x,y) &= 8(x^4 - 2x^3 + x^2)(4y^3 - 2y) \\ v(x,y) &= -8(4x^3 - 6x^2 + 2x)(y^4 - y^2). \end{aligned} \tag{4.2}$$

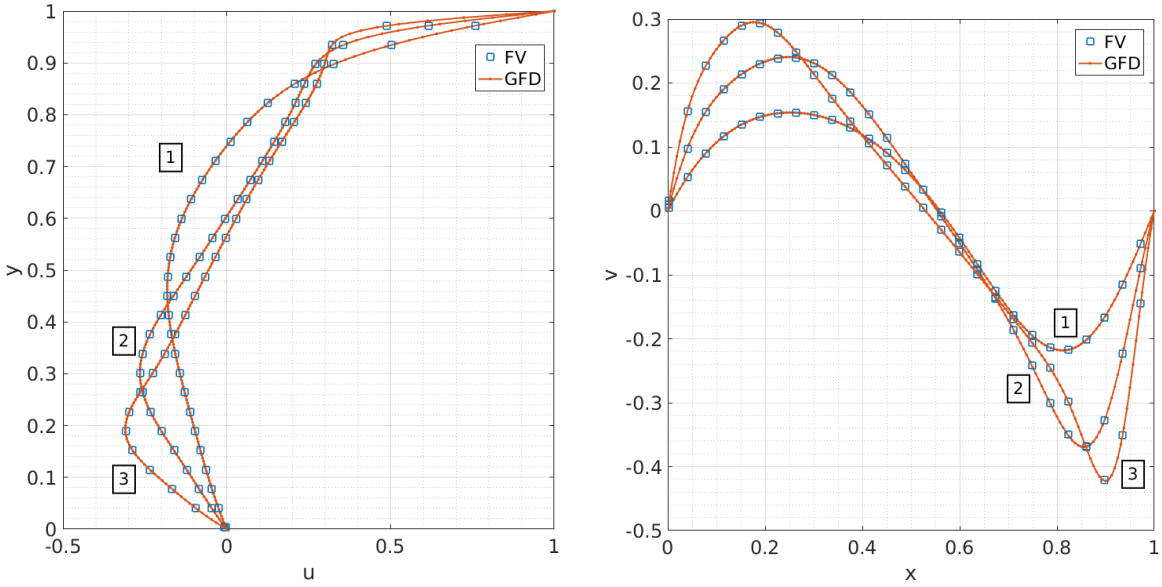


(a) BDF-1. (b) BDF-2.

Figure 4.3: Modified lid driven cavity (Case 1). L_2 relative error norm plotted at steady state for different $N \approx \sqrt{NP}$ for fixed (solid) and moving meshfree grids (dotted).

For this manufactured solution, the velocity field is independent of the Reynolds number. The lid speed for this case is $U_{lid}(x,1) = 16(x^4 - 2x^3 + x^2)$. The purpose of this test case is to evaluate the convergence order of the scheme when wall boundary conditions are imposed. The discrete relative L_2 norm is plotted at steady state for solutions obtained

using BDF-1 and BDF-2. Figure 4.3a. indicates that the BDF-1 scheme produces a solution polluted with first order splitting errors at steady state. Under diffusive time stepping ($\Delta t \propto \Delta x^2$), BDF-1 recovers approximately a second order spatial convergence rate on velocity. We find no difference in convergence rate between the fixed (solid lines) and moving grids (dotted lines). The BDF-2 error is plotted in Figure 4.3b with $\Delta t \propto \Delta x$. In addition to velocity errors, we have plotted the pressure error. On moving grids, quadratic extrapolations are not stable and so linear approximations are used for pressure, which we suspect reduce pressure convergence to first order which in turn appears to reduce the convergence of the vertical velocity component to less than second order (i.e., $O(h^{1.7})$).



(a) Horizontal velocity component along vertical centerline.

(b) Vertical velocity component along horizontal centerline.

Figure 4.4: Modified lid driven cavity (Case 2). The moving grid GFD solution ($N \approx 160$) is compared to a staggered grid solution corresponding to $NC = 161$ cells. The labels 1,2, and 3 correspond to $Re = 100$, 400, and 1000, respectively.

For Case 2, we specify a parabolic profile for the lid speed $U_{lid}(x, 1) = 4x\,(1 - x)$ and do not include any body force. This problem also has a continuous lid speed but does not have an analytic solution. BDF-2 is applied on moving meshfree grids and the solution is sampled onto a lattice using a quadratic basis. Figure 4.4 compares the GFD solution to

the staggered grid equivalent of the BDF-2 scheme wherein the continuity equation is left unperturbed [88]. The figure indicates that the GFD and FV solutions agree well throughout the range of Reynolds numbers considered.

Figure 4.5 illustrates the position of grid after an extended integration period for $Re = 400$. Initially, the nodes were marked based on whether they were above or below the horizontal centerline. As can be seen, arbitrary grid distortion is readily handled and more importantly, grid regularity is maintained throughout the integration procedure.

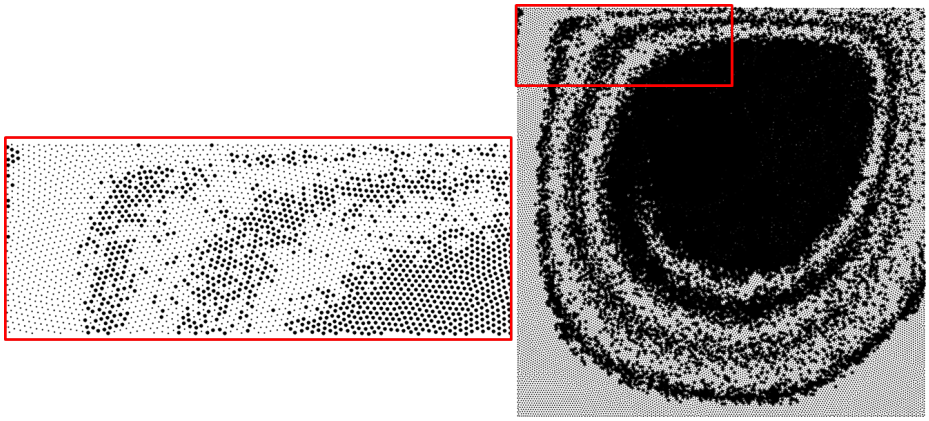


Figure 4.5: Meshfree grid at $t = 25$ for the modified lid driven cavity (Case 2) with $Re = 400$. Here, t corresponds to the time required for a unit speed lid to traverse the unit length side 25 times.

4.3 Inline oscillating cylinder

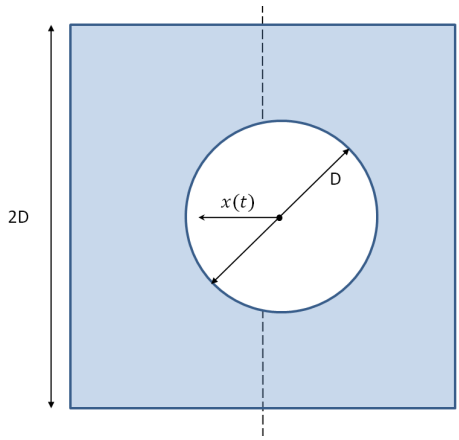


Figure 4.6: Inline oscillating cylinder computational setup. The periodic movement of a cylinder is prescribed in a closed container with stationary walls.

The final test case considered is an incompressible flow around an inline oscillating cylinder situated in a closed container with stationary walls [103]. From the method implementation point of view, for our solver there is little difference between modeling flows with and without moving boundaries. In general, as shown earlier, the incremental BDF-2 scheme should be used as it allows to use the less restrictive time stepping of $\Delta t \propto \Delta x$ while still maintaining second order accuracy. However, here we have chosen to use the non-incremental BDF-1 scheme to illustrate that the splitting error can be controlled for this more complex test case by reducing the time step.

Rigid body motion is prescribed such that center of the cylinder follows the following non-dimensional path

$$\tilde{x}(\tilde{t}) = \frac{A}{D}\left[cos\left(\frac{2\pi}{K_c}\tilde{t}\right) - 1\right] + \tilde{x}_c. \tag{4.3}$$

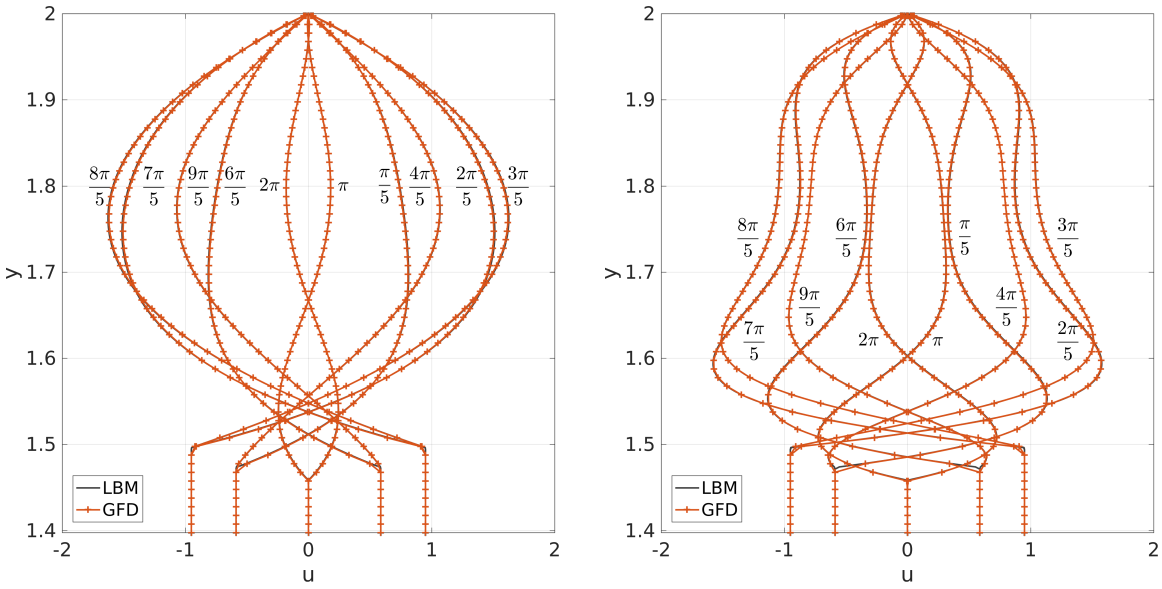
Here, the initial non-dimensional x coordinate of the center is set to $\tilde{x}_c = 1.2$, the amplitude to diameter ratio is fixed to $A/D = 0.2$, and where K_c is the Keulegan-Carpenter number $K_c = \dfrac{U_{max}}{fD}$ with U_{max} the peak velocity and f the frequency of oscillation. The horizontal velocity of the cylinder is prescribed as

$$\tilde{u}(\tilde{t}) = -\frac{2\pi A}{K_c D} sin\left(\frac{2\pi}{K_c}\tilde{t}\right). \tag{4.4}$$

We consider the case $K_c = 2\pi/5$ fixed while we vary the Reynolds number $Re = U_{max}D/\nu$ (10, 100, 400, 1000). To validate our GFD solutions, we also simulate the problem using a weakly compressible Lattice Boltzmann Method (LBM) [104–106]. Briefly, we use the D2Q9 lattice with the Bhatnagar-Gross-Krook (BGK) collision operator. Fluid is simulated both outside and inside the oscillating cylinder. Boundary conditions are enforced on the inner and outer surfaces of the cylinder surface using the linear interpolated bounce back rule [107]. We take $N = 800$ nodes per container side and perform 120,000 steps per oscillation period which for $Re = 10$, 100, 400, and 1000 results in relaxation rates of $\tau \approx 1$, 0.55, 0.505, and 0.5047, respectively. The considered Reynolds

numbers result in time periodic solutions. In all cases, the Mach number was fixed at $Ma \approx 0.007 << 1$. All simulations were executed for 10 periods in order to ensure that the solution is not affected by the initial transient.

Figure 4.7 shows that the GFD solution with approximately $N = 360$ nodes along a container side has excellent agreement with the LBM solution for both $Re = 10$ and $Re = 100$. Table A.1 in the appendix provides the tabulated horizontal velocity profiles at the different time snapshots for $Re = 10$ and $Re = 100$. For $Re = 100$, the effect of fluid inertia becomes more prominent resulting in increased gradients near the cylinder surface. The horizontal velocity profile is only shown for the top third as the solution is symmetric with respect to the horizontal mid-line. The peak cylinder speed corresponds to when the cylinder center is passing through the vertical mid-line, while minimum speed corresponds to maximum displacement (i.e., π and 2π curves).

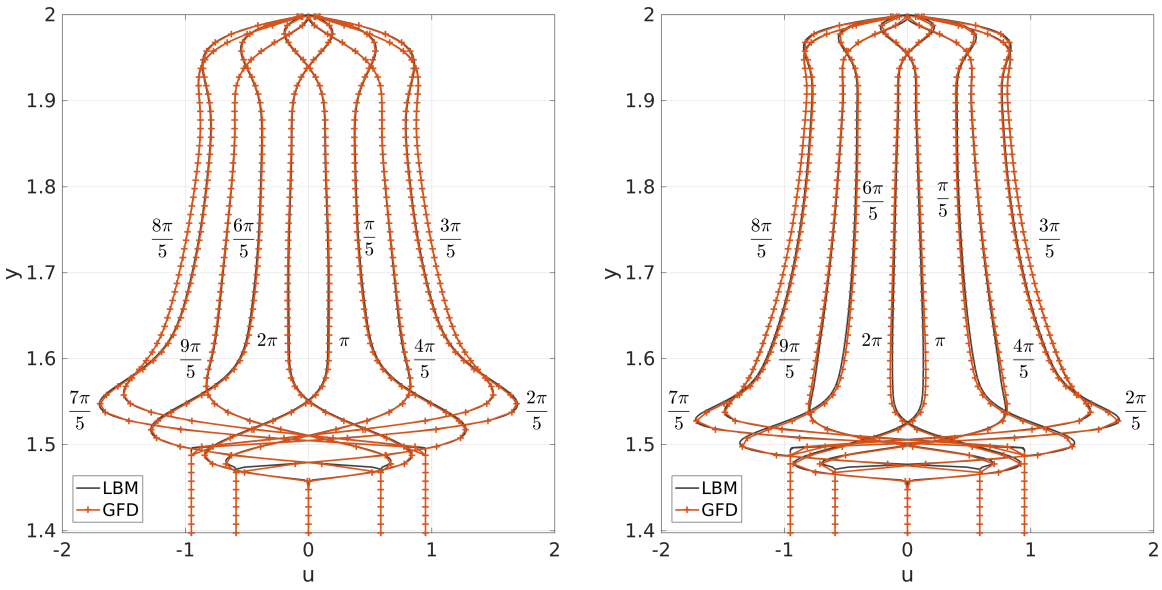


(a) $Re = 10$ (b) $Re = 100$

Figure 4.7: Oscillating cylinder horizontal velocity calculated by GFD ($N = 360$) and LBM ($N = 800$). The profiles are obtained at the tenth period and sampled every $\frac{\pi}{5}$ radians at the vertical centerline. For GFD, the Courant number $C_o \approx 0.14$ resulting in $T = 1600$ steps per period.

As shown in Figure 4.8, increasing the Reynolds number to $Re = 400$ and $Re = 1000$,

results in plug-like profiles characteristic of inertia-dominant flows. It is only near the cylinder surface and near the top wall that the solution changes rapidly, with the cylinder surface having the largest gradients. For example, the time snapshot $7\pi/5$ for $Re = 1000$ shows that near the cylinder surface the horizontal velocity component rapidly changes from approximately 0.951 to -1.713 over a small distance of $0.0175D$. Large gradients that rapidly vary become problematic for coarse resolutions since in this case the cloud radius is significantly larger than the boundary layer thickness.



(a) $Re = 400$ (b) $Re = 1000$

Figure 4.8: Oscillating cylinder horizontal velocity calculated by GFD ($N = 360$) and LBM ($N = 800$). The profiles are obtained at the tenth period and sampled every $\frac{\pi}{5}$ radians at the vertical centerline. For GFD, the Courant number $C_o \approx 0.14$ resulting in $T = 1600$ steps per period.

Table A.2 in the appendix tabulates the corresponding velocity profiles for these higher Reynolds numbers. As the Reynolds number increases so do the slight deviations between the GFD and LBM solutions. Table 4.1 quantifies the differences between the two solutions using the L_2 relative norm.

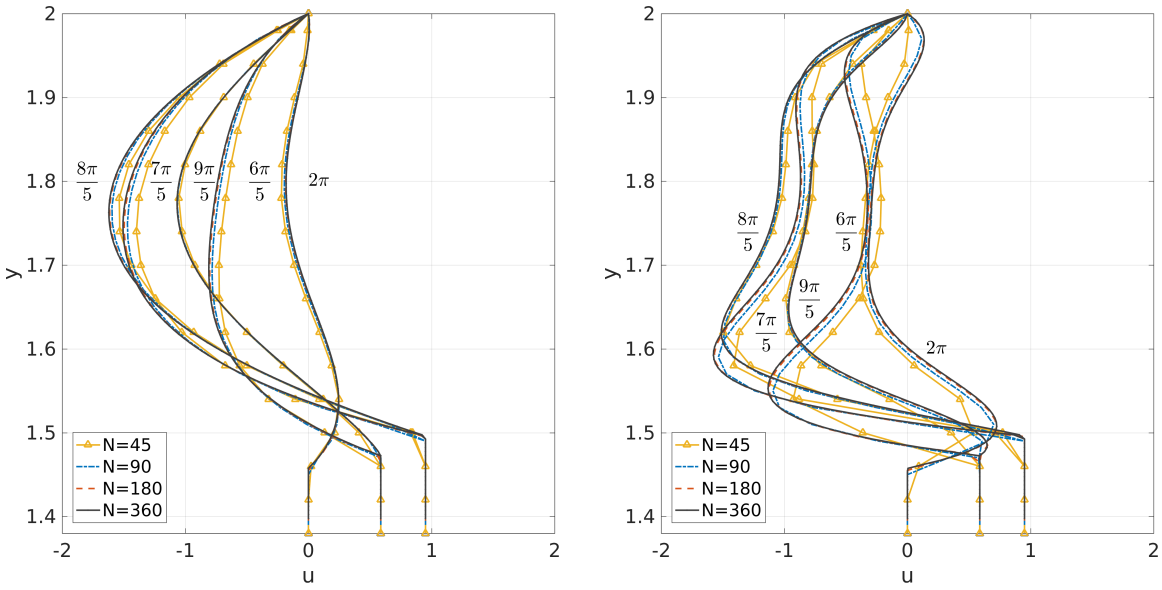
Table 4.1: L_2 relative norm for the GFD ($N = 360$) solution as compared to the LBM ($N = 800$) solution at different Reynolds number for the oscillating cylinder.

	$6\pi/5$	$7\pi/5$	$8\pi/5$	$9\pi/5$	2π	avg
$Re = 10$	7.27E-3	5.40E-3	3.68E-3	2.55E-3	1.46E-2	6.69E-3
$Re = 100$	9.63E-3	5.62E-3	2.41E-3	3.88E-3	1.33E-2	6.97E-3
$Re = 400$	1.91E-2	1.09E-2	4.16E-3	9.61E-3	3.41E-2	1.56E-2
$Re = 1000$	3.23E-2	1.90E-2	1.03E-2	1.78E-2	6.98E-2	2.98E-2

Figure 4.9 shows the GFD solution as the grid resolution increases from approximately 23 nodes across the cylinder diameter to 180 for fixed Courant number C_o. A resolution of 23 nodes across the cylinder diameter can only roughly estimate the velocity profile for $Re = 10$ and $Re = 100$. This coarse resolution corresponds to approximately 1,600 grid points while $N = 90$, 180 and 360 resolutions correspond to about 6,400, 25,800, and 103,500 fluid nodes, respectively. The convergence order p can be estimated using [108]

$$p = \frac{log\left(\frac{u^{2h} - u^{4h}}{u^h - u^{2h}}\right)}{log(2)}. \tag{4.5}$$

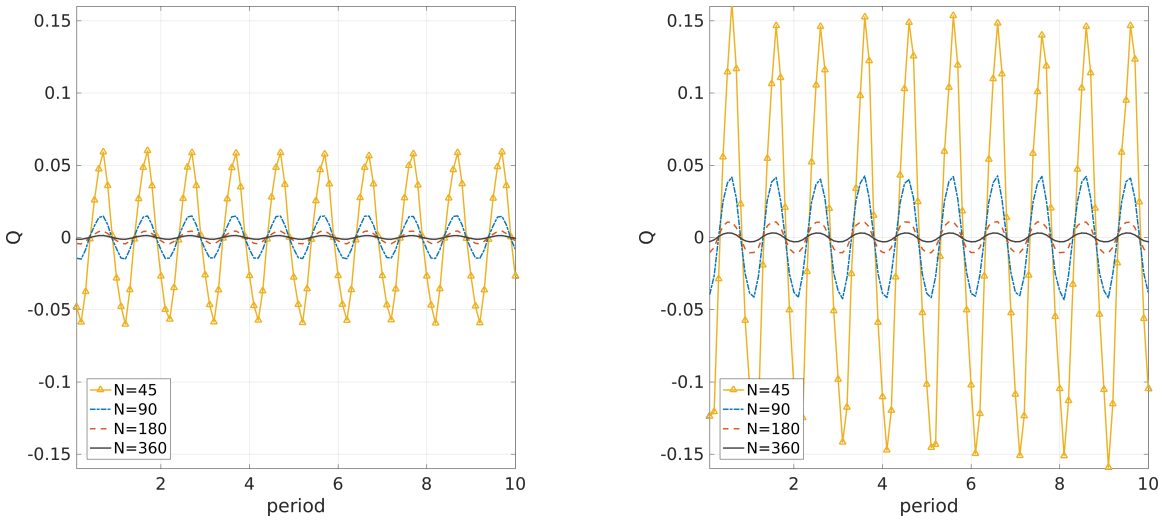
Here, u^h, u^{2h}, and u^{4h} correspond to the quadratically sampled GFD solutions for the grids $N = 360$, 180, and 90. Note that the factor of 2 in the denominator holds as we have properly increased the grid density. The convergence orders for each sampled point in the fluid are spatially and temporally averaged. For $Re = 10$, with $\Delta t \propto \Delta x^2$, BDF-1 converges with $p = 2.1$. For fixed Courant number $C_o \approx 0.14$ ($\Delta t \propto \Delta x$), BDF-1 converges with order $p = 1.7$.



(a) $Re = 10$ (b) $Re = 100$

Figure 4.9: GFD solutions for the horizontal velocity at different time snapshots for the oscillating cylinder with $Re = 10$ and $Re = 100$ at different grid resolutions N. For $N \approx 45$, 90, 180, and 360, the Courant number $C_o \approx 0.14$ resulting in $T = 200$, 400, 800, 1600 time steps per period, respectively.

Since the approximate projection method does not enforce incompressibility exactly, it is important to evaluate how well the method enforces mass conservation. To this end, we calculate the integral Q of the sampled horizontal velocity profile. The net flow rate is plotted in Figure 4.10 for different grid resolutions and two values of the Reynolds number. Note that for a truly incompressible flow $Q = 0$. The figure indicates that Q oscillates with the frequency of the cylinder oscillations and the peak values of Q converge to the expected zero value as the resolution increases for both Re. We find, however, that the magnitude of the incompressibility error is undesirably a function of the Reynolds number.



(a) $Re = 10$ (b) $Re = 100$

Figure 4.10: Time evolution of the net flow rate around the oscillating cylinder calculated across the vertical centerline for different grid resolutions using a fixed Courant number $C_o \approx 0.14$.

Table 4.2 summarizes the peak flow rate error as a function of the grid resolution and the number of time steps per period. The data in Table 4.2 indicates that for $\Delta t \propto \Delta x$, the peak flow rate error decreases between a factor of 3 and 4 for all time steps considered, therefore demonstrating a convergence rate between the first and second order rates. If instead the diffusive time step is set such that $\Delta t \propto \Delta x^2$, then the flow rate error reduces by a factor of four. This can be seen by comparing $N = 45, 90, 180$ with respectively $T = 200, 800, 3200$. In practice, the diffusive time step can be too restrictive. To bypass this limitation, the BDF-2 scheme should be used such that a fixed C_o results in second order flow rate errors.

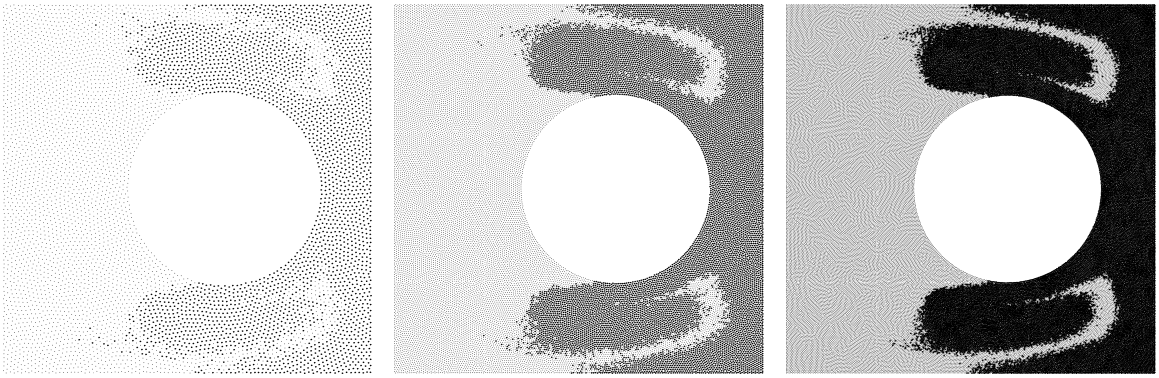
Table 4.2 also shows that the maximum Courant number of $C_o = 0.57$ was used for $N = 180$, $T = 200$, indicating that the semi-implicit approximate projection can indeed use much larger time steps than weakly compressible models. This result has been also verified for the higher Reynolds numbers. For $Re = 100$, at a coarse resolution with a small time step, the simulations were not stable. As can be seen in Figure 4.9, a resolution of $N = 45$ does not properly resolve the gradients for $Re = 100$ and so using this coarse

estimate repeatedly to update the velocity field $\mathbf{u}^{n+1}(\mathbf{x}_s)$ may be the root of the stability issues.

Table 4.2: Oscillating cylinder GFD peak flow rate error as a function of resolution and number of time steps per period for $Re = 10$ and $Re = 100$. N is approximately the number of nodes spanning the side length. Results in parenthesis were not stable.

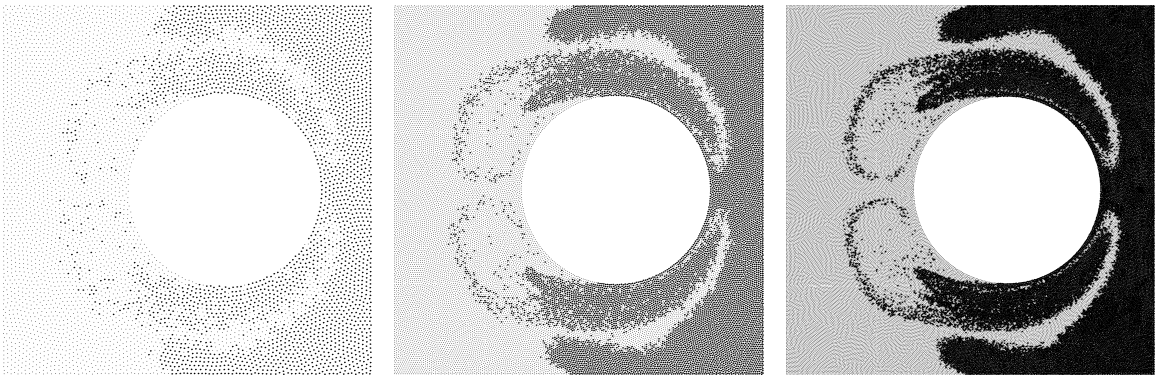
	$Re = 10$					$Re = 100$			
	T=200	T=400	T=800	T=1600	T=3200	T=200	T=400	T=800	T=1600
N=45	6.01E-2	5.16E-2	4.47E-2	4.08E-2	4.07e-2	1.63E-1	1.40E-1	(–)	(–)
N=90	1.95E-2	1.50E-2	1.15E-2	9.35E-3	7.54E-3	4.99E-2	4.31E-2	4.64E-2	4.25E-2
N=180	7.99E-3	5.88E-3	4.56E-3	3.48E-3	2.64E-3	1.68E-2	1.33E-2	1.09E-2	8.71E-3

The final positions of the grid for $Re = 10$ and $Re = 100$ are shown in Figure 4.11 and Figure 4.12 for different resolutions. Initially, nodes were marked according to whether they were located to the left or to the right of the vertical centerline. Interestingly, the grid regularization appears to maintain this artificial interface with features refined as the grid resolution is increased.



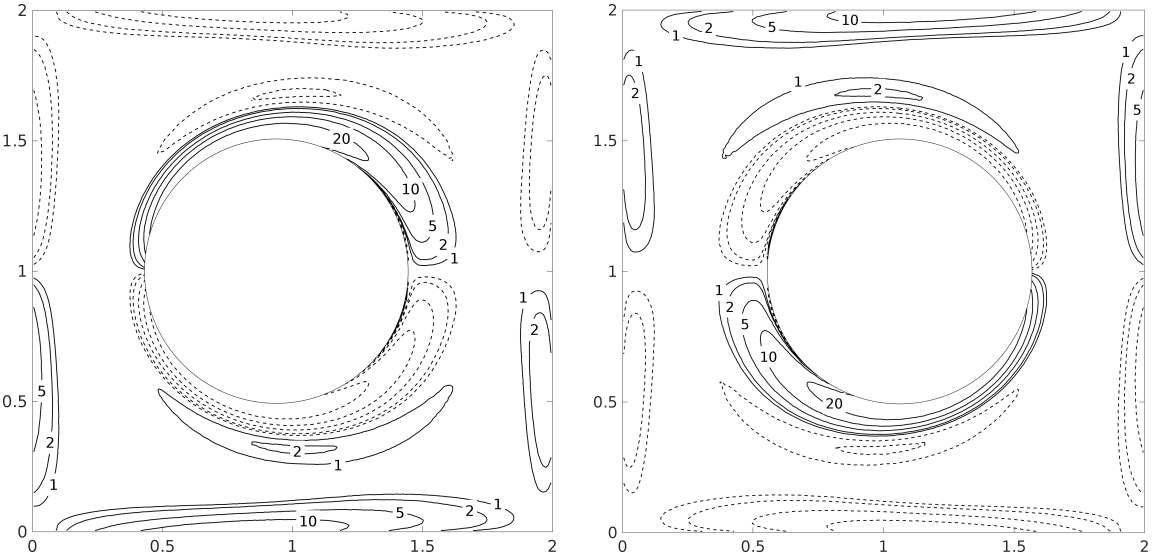
(a) N = 90 (b) N = 180 (c) N = 270

Figure 4.11: GFD meshfree grids at the end of tenth cycle for the cylinder oscillating with $Re = 10$ and different grid resolutions N.



(a) N = 90 (b) N = 180 (c) N = 270

Figure 4.12: GFD meshfree grids at the end of tenth cycle for the cylinder oscillating with $Re = 100$ and different grid resolutions N.



(a) Vorticity isolines at $3\pi/5$ radians into the tenth period.

(b) Vorticity isolines at $8\pi/5$ radians into the tenth period.

Figure 4.13: Inline oscillating cylinder vorticity isolines for $Re = 100$ during the 10th period. Negative values of vorticity are denoted by dotted lines.

Figure 4.13a. plots selected vorticity isolines for $Re = 100$ as the cylinder decelerates while moving to the left, whereas Figure 4.13b. plots the same isolines as the cylinder decelerates while moving to the right. The time snapshots are chosen π radians apart to highlight any asymmetries. As can be seen, the vorticity magnitude is qualitatively symmetric both spatially (with respect to the horizontal axis) and temporally (with respect to a shift of

π radians). Figure 4.14 plots selected pressure isolines. As can be seen, the pressure field is smooth despite pressure and velocity unknowns co-located at the grid nodes, indicating the additional pressure modes typically observed for co-located variable arrangements are eliminated. Corresponding animations are provided in the supplementary material [6].

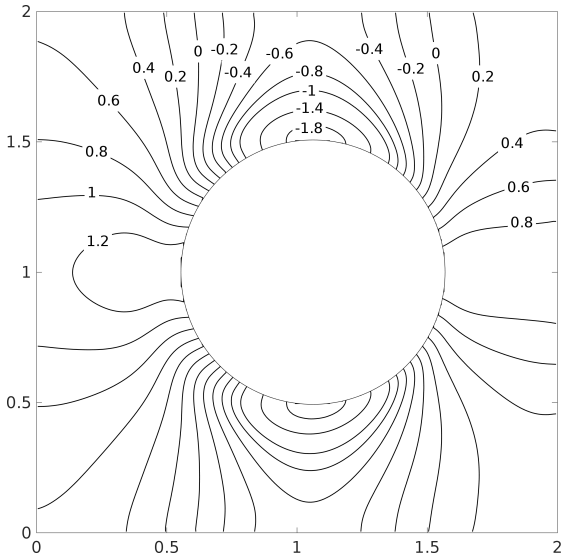


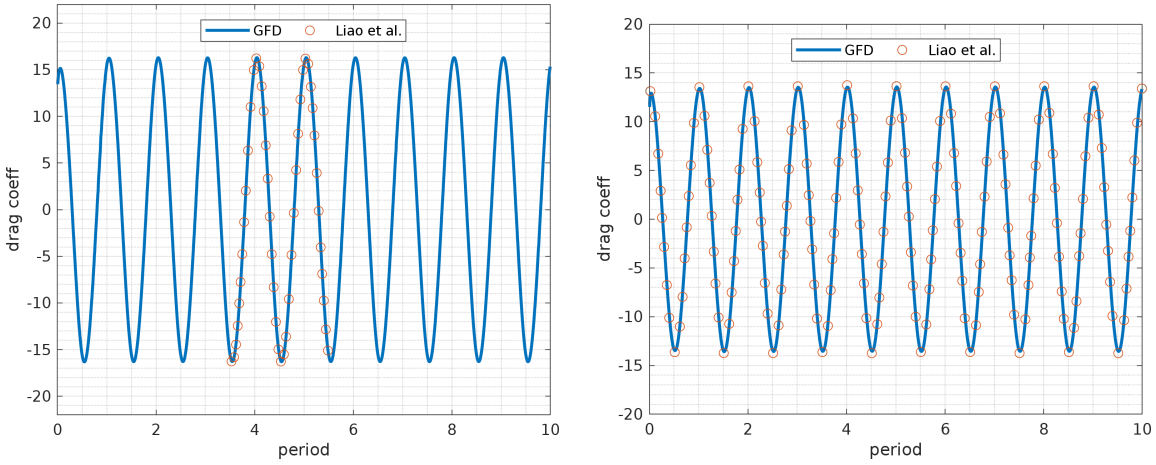
Figure 4.14: Inline oscillating cylinder pressure isolines for $Re = 100$ at $8\pi/5$ during the 10th period. Note the solution is symmetric about the horizontal axis.

To cross-validate results against literature, in Figure 4.15 we compare the drag coefficient $C_d = F_x/\left(0.5\rho U^2 D\right)$ to the immersed boundary results of Liao et al. [103]. Here, $\rho = 1$ is the fluid density, $U = 1$ is the max cylinder speed, $D = 1$ is the cylinder diameter with unit depth, and F_x is the horizontal force on the cylinder. The force is computed by directly integrating the stress tensor σ over the cylinder surface

$$F_x = -\hat{i} \cdot \int_{cyl} \sigma \cdot \hat{n} ds. \tag{4.6}$$

This surface integral is approximated by assuming a linear variation in each boundary element. The stress tensor at the cylinder surface is approximated using GFD. Note that Liao et al.'s drag force - presented in their Figure 4 and Figure 8 - includes the horizontal force on the cylinder due the fictitious mass of fluid accelerating inside the cylinder. As such,

we first remove this fictitious force and then normalize their reported drag force to a drag coefficient using their respective parameters $\rho = 1, D = 0.5, U = 0.2\pi$. As can be seen, good agreement is observed for both $Re = 100$ and $Re = 800$ with the drag coefficient smoothly oscillating at the frequency of the cylinder. This suggests that our approach can also be used when the rigid body motion is not prescribed but is instead dependent on the hydrodynamic forces.



(a) $Re = 100$ (b) $Re = 800$

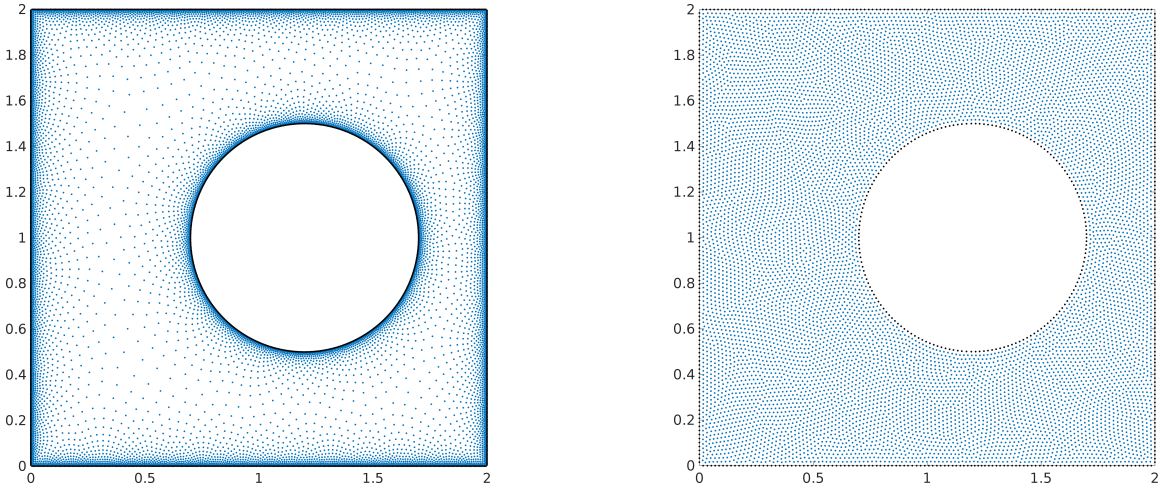
Figure 4.15: Inline oscillating cylinder drag coefficient comparison to Reference [103].

CHAPTER 5

ADAPTIVE MOVING MESHFREE GRIDS

In this section, we use the direct ALE method detailed in Section 2.5.3 to simulate incompressible flows on adaptive moving meshfree grids. Specifically, the BDF-2 scheme is used with non-incremental pressure. Three benchmarks are validated against experimental and numerical results. Namely, the inline oscillating cylinder for a small and large domain and the two moving cylinders problem.

5.1 Inline oscillating cylinder revisited



(a) (b)

Figure 5.1: (a) Oscillating cylinder benchmark and the two corresponding grids used: (b) adaptive meshfree grid and (c) quasi-uniform meshfree grid, both of which have approximately 10,000 points.

We revisit the inline oscillating cylinder and consider the case of $H/D = W/D = 2$, with $K_c = 2\pi/5$ while varying the Reynolds number $Re = U_{max}D/\nu$ (10, 100, 400, 1000). As shown in Figure 5.1, we consider an adaptive meshfree grid and a quasi-uniform meshfree grid which are comprised of approximately the same number of total points (i.e., 10,000).

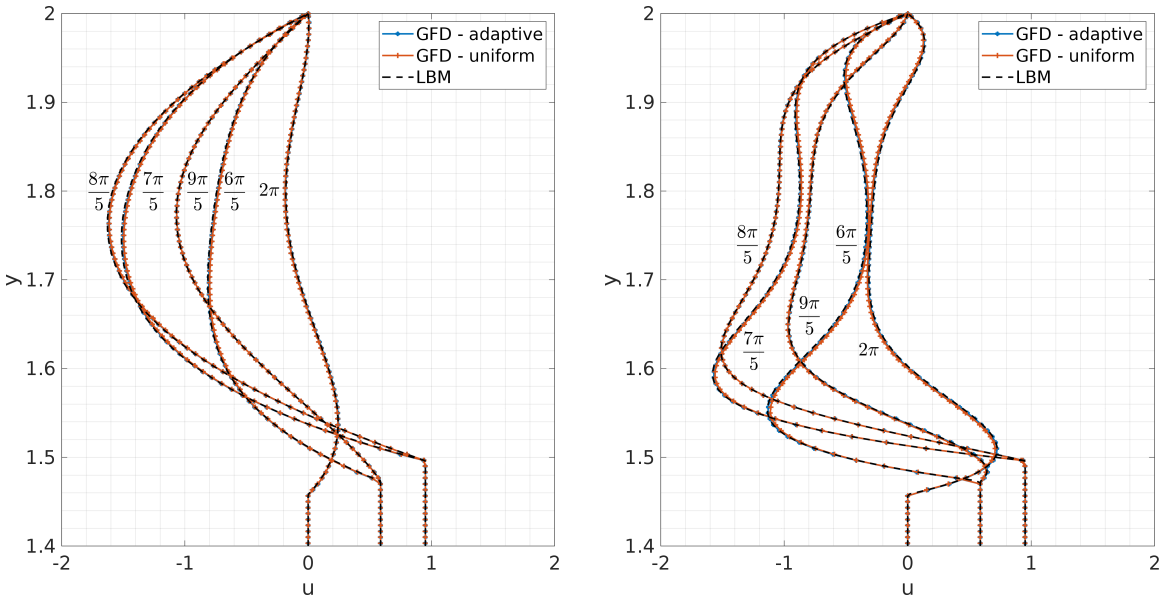
Both grids are generated using the approach detailed in Section 2.6 with the only difference being that in the quasi-uniform grid the spring equilibrium length was constant everywhere, whereas for the adaptive grid the spring equilibrium length was allowed to vary based on the distance function to the surfaces.

For the adaptive grid we choose a clustering ratio of $\Delta x_{max}/\Delta x_{min} = 15$ in Eq. 2.95. With an initial 400×400 lattice, this results in a grid comprised of roughly 10,000 nodes where several points (3-4) span the thickness of the thin layer of fluid near the cylinder surface for the highest Reynolds number case. The grid is shown in Figure 5.1a. By targeting the highest Reynolds number case, we are able to reuse the same grid for the lower Reynolds number cases, although there will be more nodes than required to sufficiently resolve the solution in the respective boundary layers. The clustering ratio $\Delta x_{max}/\Delta x_{min} = 15$ was also partly chosen as it is approximately the max clustering ratio that can be used for $Re = 1000$ with $\Delta x_{min} \approx 1/200$ without instabilities arising. If larger ratios are used, the transition from Δx_{max} to Δx_{min} will be covered by too few grid points. This observation is analogous to the mesh-based criteria that requires neighboring element sizes should not exceed each other by more than a certain factor. In other words, the magnitude of the gradient of the spring equilibrium length must be limited. In our case, gradient limiting can be controlled directly by reducing $\Delta x_{max}/\Delta x_{min}$ or h_0 in Eq. 2.95, both of which will increase the number of grid points in the domain. More generally, the gradient limiting equation can be solved [109].

In Figure 5.2 and Figure 5.3, we sample the horizontal velocity component along the vertical centerline for both quasi-uniform and adaptive grids and compare to our previous high resolution LBM results [6] for $Re = 10, 100$ and $Re = 400, 1000$, respectively. The grid velocity has been set as $\hat{\mathbf{u}} = \hat{\mathbf{u}}_1 + \Delta \mathbf{x}_s/\Delta t$, where $\hat{\mathbf{u}}_1$ is the solution to Laplace's equation with appropriate boundary conditions and where $\Delta \mathbf{x}_s$ is the spring-based component given by Eq. 2.93. For the adaptive grid, excellent agreement is observed throughout the range of Reynolds number using only a fixed $n = 13$ nearest neighbors per grid point. Re-

call from earlier that most SPH solvers require several times this many neighbors per node. The low neighbor count used here was only possible due to the consistent GFD spatial discretization and our high quality grids.

Additionally, the time step used here results in a Courant number $C_o = \Delta t U/\Delta x_{min} \approx 0.31$ on the adaptive grid, where $T = 800$ is the number of steps per period, $\Delta t = K_c/T$ and Δx_{min} are the non-dimensional time step and grid spacing with the reference speed chosen as the max cylinder speed. This time step is not possible with the weakly compressible adaptive SPH solvers. The relatively large time step and the low neighbor count are critical to offsetting the computational burden imposed by solving multiple linear systems every time step.

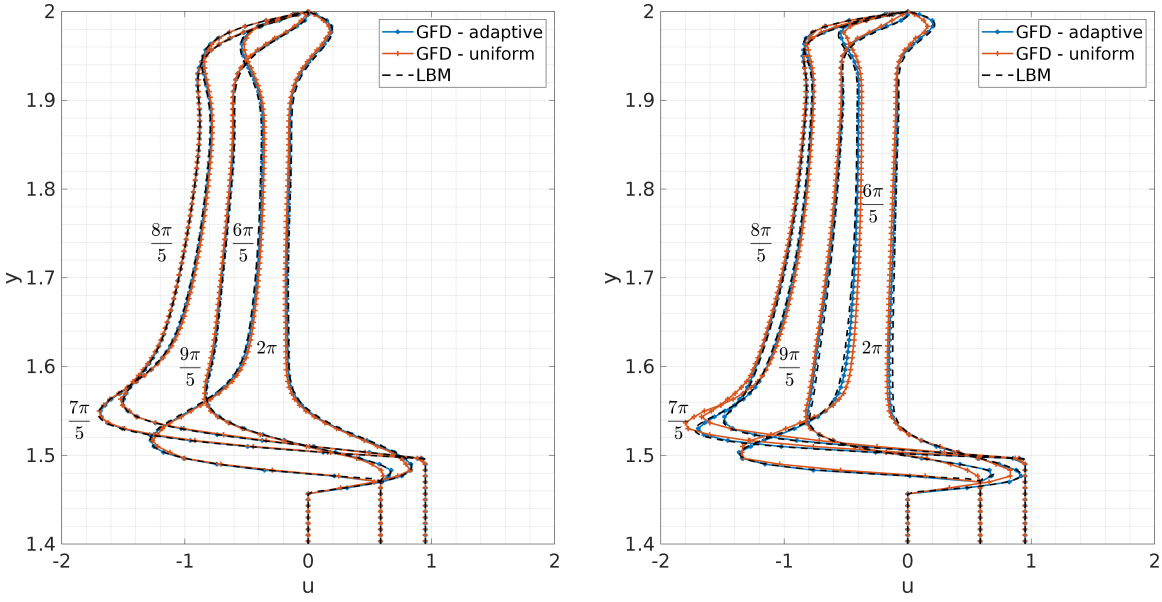


(a) $Re = 10$ (b) $Re = 100$

Figure 5.2: Oscillating cylinder horizontal velocity calculated by GFD and LBM ($N = 800$). The profiles are obtained at the twentieth period and sampled every $\frac{\pi}{5}$ at the vertical centerline. The quasi-uniform and adaptive GFD solutions are comprised of roughly 10,000 points with $T = 800$ steps per period used in both cases.

The same problem using the quasi-uniform meshfree solver requires approximately 8 times the number of grid points to obtain the same resolution Δx_{min} near the cylinder surface. However, as can be seen by the overlapping of the adaptive and quasi-uniform grid

($\Delta x \approx 0.017$) solutions, the fine resolution obtained by the adaptive grid is not needed to adequately capture the low Reynolds number flow where the gradients are not as sharp and vary slowly (i.e., $Re < 100$). As the Reynolds number increases, the quasi-uniform resolution does not fully resolve the spatial variations in the solution. For these higher Reynolds numbers, the quasi-uniform grid solution deviates from the LBM solution considerably within the thin layer of fluid adjacent to the cylinder surface, whereas the adaptive grid solution still matches the LBM solution well. However, while the geometric adaptivity works well enough to illustrate the method, it is by no means optimal. Indeed, a better clustering strategy would be to redistribute, insert, or remove points based on estimates of the truncation error. This is significantly more complex and is not addressed here.

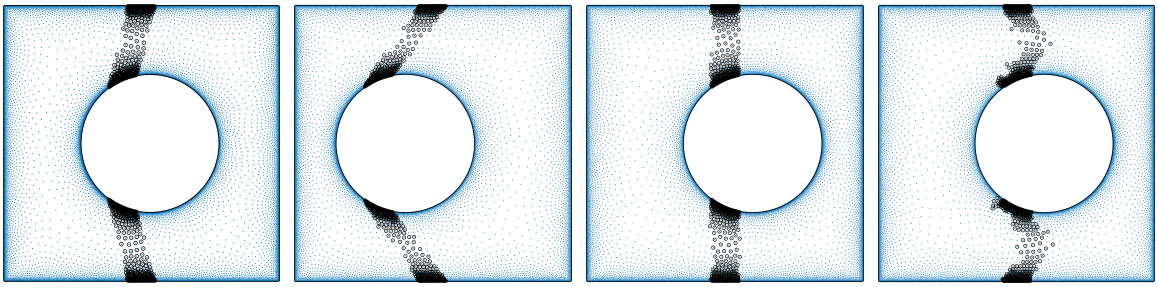


(a) $Re = 400$ (b) $Re = 1000$

Figure 5.3: Oscillating cylinder horizontal velocity calculated by GFD and LBM ($N = 800$). The profiles are obtained at the twentieth period and sampled every $\frac{\pi}{5}$ at the vertical centerline. The GFD quasi-uniform and adaptive grid solutions are comprised of roughly 10,000 points with $T = 800$ steps per period used.

Figure 5.4 illustrate the resulting grid motion for the adaptive grid. Initially, grid points within $0.9 < \tilde{x} < 1.1$ are marked (i.e., the dark vertical band). During the first few cycles the dark vertical band exhibits a periodic shearing behavior due to the grid sticking

to the moving and stationary surfaces. Over time, a single iteration for the spring-based component locally spreads grid nodes apart while maintaining the specified adaptivity. As a result, at the end of the twentieth cycle (i.e., Figure 5.4d) the dark patch of nodes is no longer vertical as it would be if the grid motion was solely determined by the Laplacian-based component. Moreover, the grid motion shown here is identical for all Reynolds numbers as this Laplacian-based grid motion is decoupled from the Reynolds number for prescribed rigid body motion.



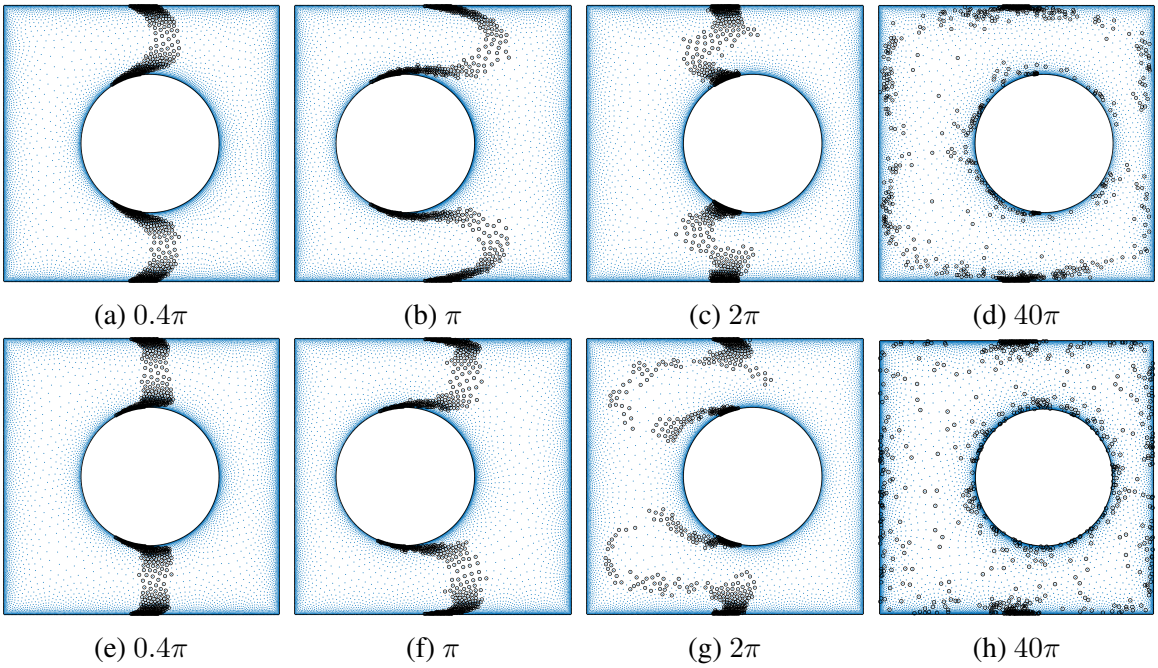
(a) 0.4π (b) π (c) 2π (d) 40π

Figure 5.4: Grid movement throughout the first cycle $[0, 2\pi]$ (a-c) and final grid distortion (d) using a grid velocity constructed from a Laplacian-based component and a spring-based component.

Throughout the 20 periods considered, a high average grid quality of $q_{avg} > 0.9$ is maintained with the minimum quality oscillating between $0.5 < q_{min} < 0.7$ throughout. If the spring-based component is dropped, the grid quality deteriorates within the first cycle leading to the development of instabilities. Similar observations are made if the grid is strictly evolved along the particle path (i.e., pure Lagrangian).

Alternatively, in the direct ALE solver the grid velocity can be specified as $\hat{\mathbf{u}} = \mathbf{u} + \Delta\mathbf{x}_s/\Delta t$, where $\mathbf{u}$ is the physical fluid velocity. Using this Lagrangian-based motion is analogous to the 3-stage ALE schemes commonly used in SPH, where with the direct ALE approach the rezoning and remapping stages are merged into the position integration and momentum solve, respectively. The advantage of the approach is it is cheap, moves nodes naturally around obstacles, and maintains a smaller convective term. However, as shown in Figure 5.5 larger grid distortions are observed. The grid motion is illustrated for $Re = 10$

and $Re = 100$ respectively, where note the motion now is a function of the Reynolds number. Due to the interplay between fluid inertia and viscous forces, grid distortion is increased as the Reynolds number is increased. It is clear that for a mesh-based ALE method, both cases would require mesh surgery, whereas this is entirely avoided with a meshfree direct ALE approach.



(a) 0.4π (b) π (c) 2π (d) 40π

(e) 0.4π (f) π (g) 2π (h) 40π

Figure 5.5: Grid movement throughout the first cycle (a)-(c) and final grid distortion (d) for $Re = 10$ and $Re = 100$ (e)-(h) using a grid velocity constructed as the sum the Lagrangian-based component and spring-based component.

Depending on the Reynolds number, the Lagrangian-based grid motion may require multiple iterations of Eq. 2.93 to maintain a high quality grid, whereas the Laplacian-based motion required only a single iteration for this test case irrespective of the Reynolds number. For instance, for $Re = 10$, the Lagrangian-based approach required a single iteration to maintain a high grid quality, whereas for $Re = 100$ the time step was reduced by half (i.e., $T = 1600$ steps per period) and four iterations of Eq. 2.93 were required to maintain grid regularity throughout the 20 periods considered. Using too few iterations leads to instabilities rooted in irregular point cloud distributions. With this in mind, in the

following two sections, only the first approach (i.e., Laplacian-based motion) is used to evolve the grids.

5.1.1 Large domain

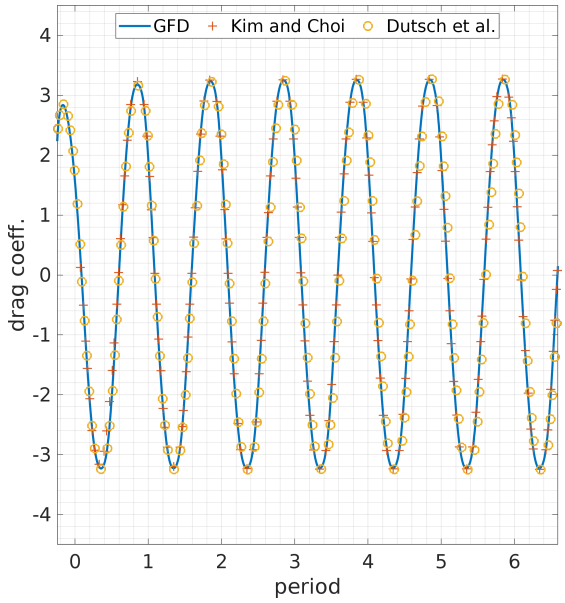


Figure 5.6: Drag coefficient comparison to Dutsch et al.'s experimental results [110] and Kim and Choi's immersed boundary results [111] for an inline oscillating cylinder in an unbounded domain at $Re = 100$ and $K_c = 5$.

Using adaptivity it is possible to consider larger domain sizes while keeping the computational time reasonable. Specifically, we again consider the case of an inline oscillating cylinder with the following parameters: Keulegan - Carpenter number $K_c = 5$ and Reynolds number $Re = 100$. For this set of non-dimensional numbers, Dutsch et al. [110] report experimental and numericals results. In addition, Kim and Choi report immersed boundary numerical results [111].

The computational grid is similar to the previous section except now we increase the ratio of the box dimensions to diameter by an order of magnitude to $H/D = W/D = 20$. Moreover, no clustering is used near the distant boundaries. To evolve the grid, the grid velocity is constructed as the sum of the Laplacian-based component and spring-based

component. The target resolution is set to be $\Delta x_{min} \approx 1/50$ (i.e., approximately 50 points across the diameter). The clustering ratio is set to $\Delta x_{max}/\Delta x_{min} \approx 20$ in Eq. 2.95 and distance based adaptivity to the cylinder surface is set. These parameters result in a initial grid with approximately 21,000 grid points which is then evolved in time based on the specified grid velocity. The time step is set such that $T = 1600$ time steps per oscillation period are taken resulting in a Courant number $Co \approx 0.16$ on the minimum grid spacing.

In Figure 5.6, the results from the direct ALE approach for the drag coefficient are compared to experimental and numerical results. As we can see good agreement is observed with our results matching more closely with Kim and Choi's numerical results [111].

5.2 Two moving cylinders

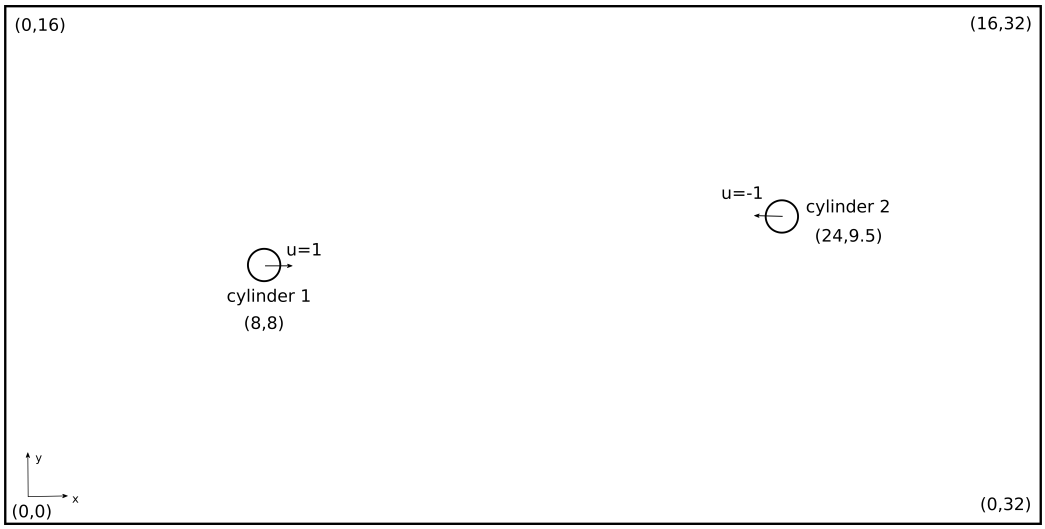


Figure 5.7: Two moving cylinders problem.

Using the direct ALE approach we now consider the more challenging two moving cylinder problem [112]. As shown in Figure 5.7, two cylinders are placed sixteen diameters apart in an enclosed stationary container and begin to move towards each other at constant speed in an incompressible fluid with Reynolds number of $Re = 40$. The vertical position of the cylinder centers are set one and half diameters apart so that as the cylinders approach they

avoid collision by a vertical gap exactly the size of half the diameter of the cylinder. Within a close distance the two cylinders interact and impart forces on each other through the fluid.

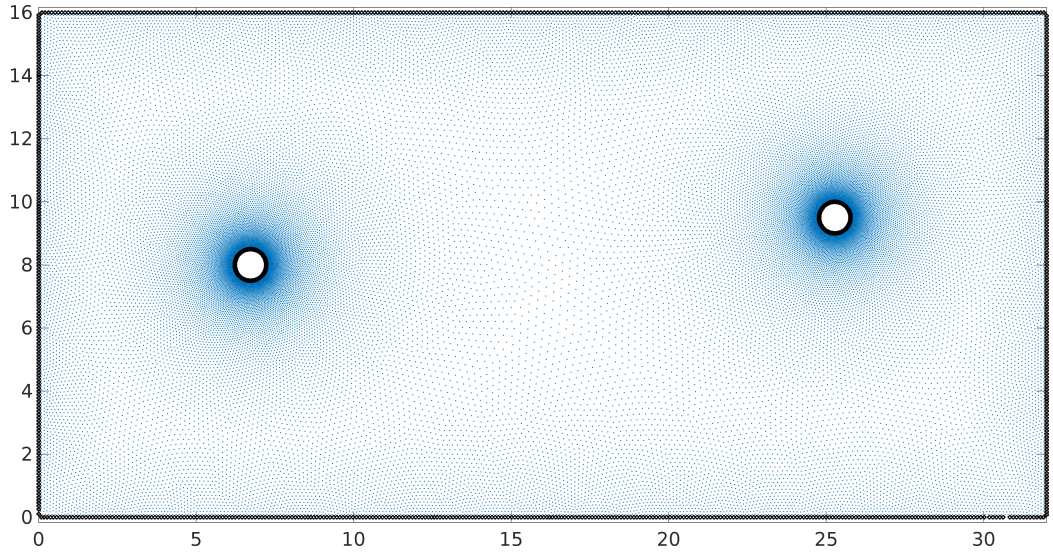


Figure 5.8: Example computational grid at $t = -2$ for the two moving cylinders problem. The maximum to minimum spacing is approximately ten to one with approximately 42,000 total grid points.

Similar to Xu and Wang [113], before moving towards each other, we let each cylinder oscillate for two periods to avoid the impulsive start. Specifically, each cylinder is prescribed the following horizontal motion

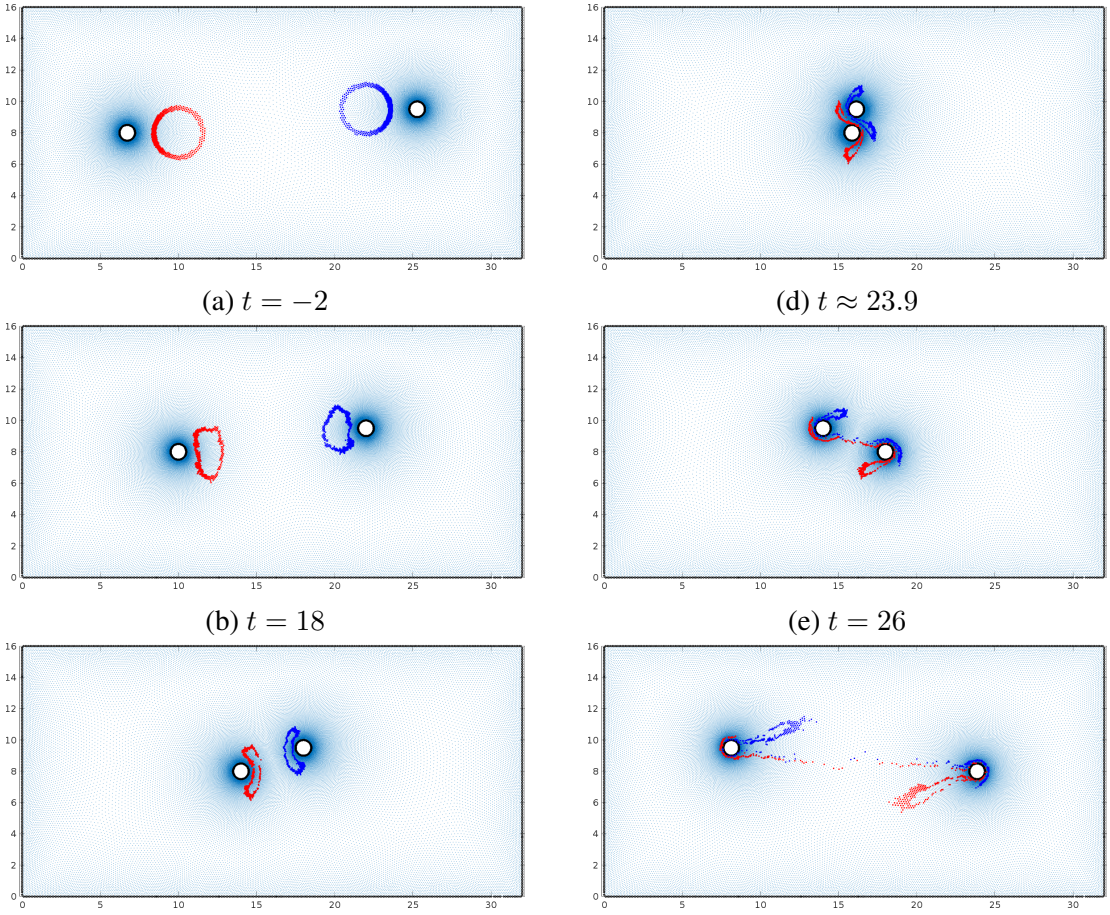
$$\text{lower cylinder}: \begin{cases} x_c(t) = 8 + \dfrac{4}{\pi}\sin\dfrac{\pi t}{4}, & t < 16 \\ x_c(t) = 8 + (t - 16), & 16 \le t \le 32 \end{cases} \tag{5.1}$$

$$\text{upper cylinder}: \begin{cases} x_c(t) = 24 - \dfrac{4}{\pi}\sin\dfrac{\pi t}{4}, & t < 16 \\ x_c(t) = 24 + (16 - t), & 16 \le t \le 32. \end{cases} \tag{5.2}$$

Time integration is started at $t = -2$ such that cylinders initially have zero velocity. A fixed time step is set such that the Courant number $Co = 0.1$ on the minimum grid spacing. The meshfree grids are constructed with clustering near the cylinder surfaces as shown in the example grid in Figure 5.8. Here, the target resolution is $\Delta x_{min} \approx 2/75$ with a clustering

ratio of $\Delta x_{max}/\Delta x_{min} = 10$.

To decouple the grid motion from the flow, the grid velocity is constructed as the sum of the Laplace-based velocity and spring-based velocity. The Laplacian-based component results in the bulk horizontal motion of the grid nodes near the cylinders whereas the spring-based component is responsible for maintaining adaptivity and all other motion. The grid motion is shown in Figure 5.9.



(a) $t = -2$ (d) $t \approx 23.9$

(b) $t = 18$ (e) $t = 26$

(c) $t = 22$ (f) $t \approx 32$

Figure 5.9: Grid deformation for two moving cylinder problem. A patch of tracked nodes (initially in shape of an annulus) are marked near the lower cylinder (red) and upper cylinder (blue).

Here, red and blue patches of nodes are marked to track the grid deformation. Initially, these nodes each form an annulus and undergo contraction and expansion as the cylinders push and pull the grid towards and away from them (not shown). Once the cylinders exit the

oscillatory period ($t > 16$), the shape of the patches deform and closely resemble deforming bubbles. After the cylinders pass each other, these patches undergo severe deformation and tear.

Figure 5.10 highlights the critical moment as the cylinders pass each other. As we can see, the grid is still quite regular while simultaneously adaptive. Without such regularity, instabilities would quickly arise. Unlike in the previous inline oscillating cylinder examples, we observe applying only a single iteration of Eq. 2.93 is insufficient to evolve the grid around the cylinders in a regular manner while adhering to distance based adaptivity. More specifically, for the specified Courant number and spatial resolution, as many as ten to twenty iterations are required to compute a shift that maintains smooth grid motion as the cylinders begin to pass each other.

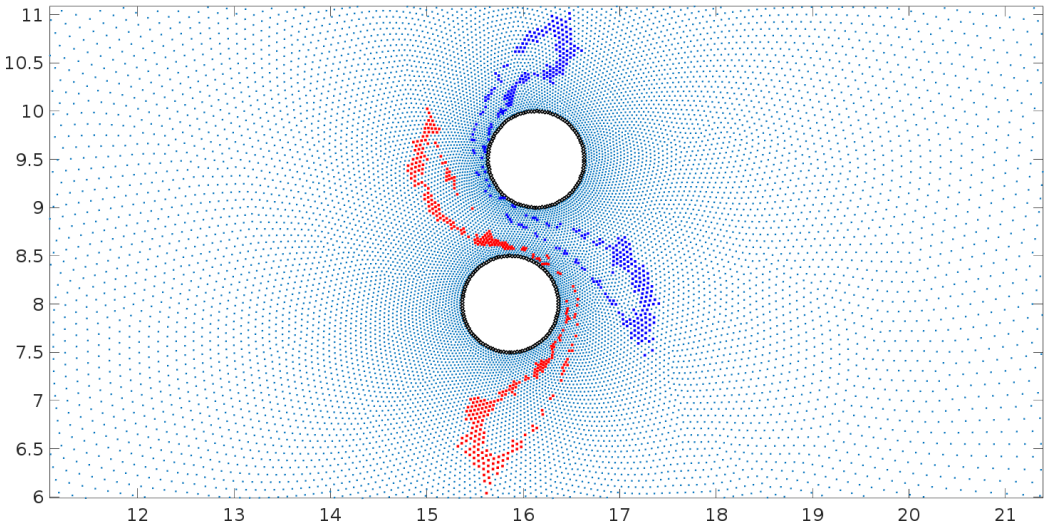


Figure 5.10: Computational grid at $t \approx 25.9$ as the two cylinders pass each other.

Figure 5.11 plots the temporal evolution of the drag and lift coefficient for the upper cylinder using a coarse grid computation comprised of approximately 22,000 nodes and a ten to one grid spacing ratio. With a target resolution of 25 nodes across the diameter and a fixed Courant number of $Co = 0.1$, the resulting time step is $\Delta t = 0.004$ with 8500 total time steps. For $t < 16$, the drag coefficient corresponds to what one would expect of an inline oscillating cylinder for the specified Reynolds number and non-dimensional

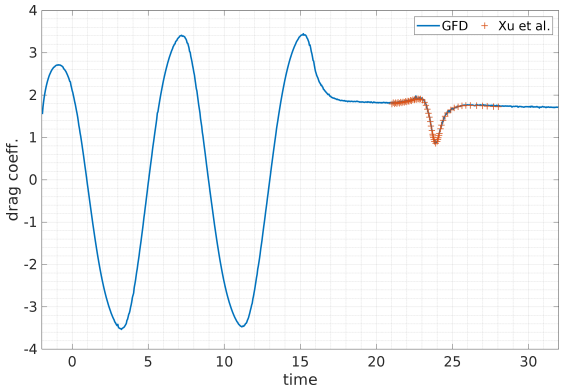
period. Similarly, as the two cylinders are far apart, the lift coefficient is approximately zero. For $t > 16$ the cylinder maintains unit speed and the drag coefficient decreases towards approximately what one may expect for uniform flow over a cylinder in a finite sized domain.

When the upper and lower cylinder centers are within six diameters of each other they begin to interact. Initially, the drag and lift on the upper cylinder increases with the drag force reaching a local maximum when the cylinder centers are approximately two and half diameters apart, whereas the lift force reaches a global maximum when the cylinder centers are approximately a diameter apart. Afterwards, as the cylinders near, the drag and lift suddenly drop. The drag force reaches a local minimum slightly before the two cylinders are directly overhead whereas the lift force reaches a global minimum when the two cylinders have passed and are separated by approximately 1.25 diameters. The drag and lift then recover with the lift force oscillating. This same behavior has been reported by others [112, 113] as well.

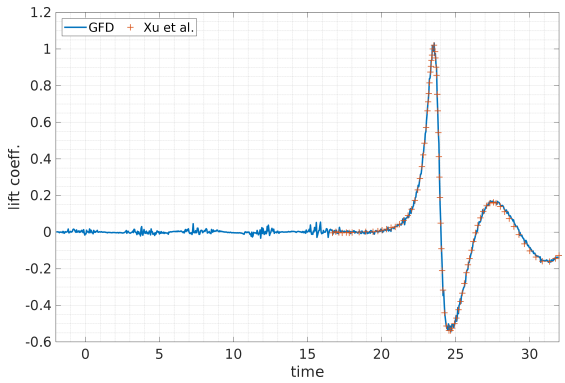
For this coarse grid computation, Figure 5.11 shows that nonphysical wiggles are present near the time points corresponding to half the oscillation period (e.g., $t = 0, 4, 8, 12$ and $t = 16$). These high frequency oscillations are more pronounced in the lift coefficient and correspond to peak cylinder speed. While the source of these oscillations is uncertain, we suspect these oscillations are due to a combination of 1) the non-linear convective term and its central difference-like discretization and 2) the non-conservative nature of our discretization.

It is observed that increasing the grid resolution and decreasing the time step or decreasing the Reynolds number will decrease the severity of the oscillations. This is confirmed in Figure 5.12 which plots the drag and lift coefficients computed on a fine grid with a reduced time step and target resolution of $\Delta x_{min} = 1/100$. Here, the x-axis is reported in terms of distance between cylinder centers in order to give a better sense of the proximity of the two cylinders. Additionally, we plot the forces on the lower cylinder, confirming they are

approximately equal and opposite. For this fine grid, the drag and lift match well with Xu and Wang's immersed interface method results [113].

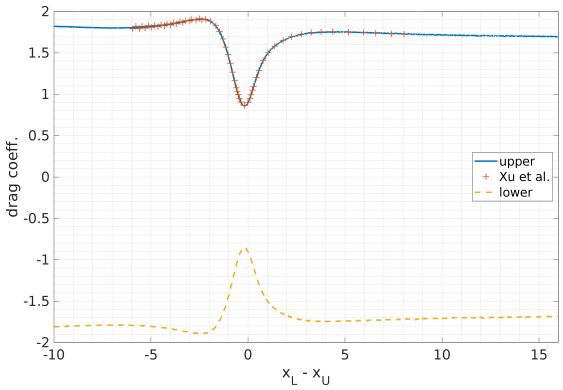


(a) Drag coefficient

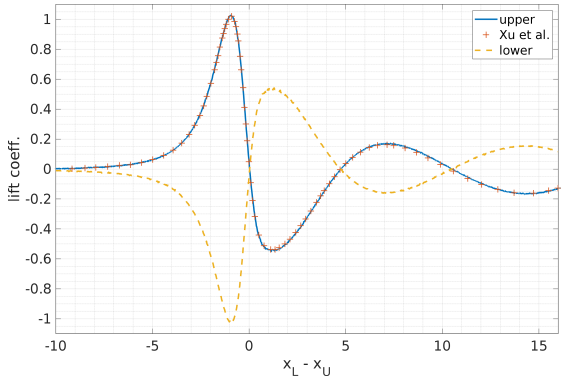


(b) Lift coefficient

Figure 5.11: Coarse grid results (22,000 nodes) with $Re = 40$. GFD drag (a) and lift (b) on upper cylinder compared to Xu and Wang [113] for the two moving cylinder problem. The Courant number is fixed to $Co = 0.1$ resulting in 8,500 time steps taken with $\Delta t = 0.004$. Drag and lift computed every tenth time step.



(a) Drag coefficient.



(b) Lift coefficient.

Figure 5.12: Fine grid (270,000 nodes) results with $Re = 40$. GFD drag (a) and lift (b) on the upper (solid line) and lower cylinders (dotted line) focused around the time interval during which the cylinders interact. The x-axis is the distance between the lower and upper cylinder centers. Results are compared to Xu and Wang [113]. The Courant number is fixed to $Co = 0.1$ resulting in 34,400 time steps taken with $\Delta t = 0.001$. Drag and lift computed every tenth time step.

CHAPTER 6
CONCLUDING REMARKS

6.1 Thesis contributions

The contribution of this thesis is a collocated, arbitrary Lagrangian-Eulerian (ALE) meshfree solver for viscous incompressible flows involving moving interfaces and adaptive refinement. The solver is purely meshfree and no background meshes are used. Instead of using a weakly compressible assumption, the incompressible Navier-Stokes equations are solved using a semi-implicit approximate projection method. Correspondingly, the acoustic and diffusive time step constraints are removed and transient solutions do not exhibit Mach number dependence.

Notably, the grid velocity was chosen to be decoupled from the fluid flow. Consequently, the grid motion evolves independently of the Reynolds number. This is in contrast to three stage SPH solvers, where the fluid flow is first solved along the Lagrangian path with the grid subsequently rezoned and fluid properties remapped. While our grids still undergo severe deformation, the degree of deformation is less pronounced than the Lagrangian-based approach. This grid velocity was constructed using a Laplacian-based component and spring-based component. The former results in bulk motion near moving interfaces, whereas the latter component is primarily responsible for controlling the grid's regularity and adaptivity.

The spring-based component was computed in a meshfree fashion using the explicit relaxation equation from Persson's simple mesh-based grid generator. This spring-based approach provides a simple framework for meshfree grid generation. With this approach, complex boundaries are represented as the zero level set and grid adaptivity is controlled through the specified spring equilibrium length. The generated meshfree grids have a

smoothly varying resolution and are of high quality. As a result, a simple "k nearest neighbor" stencil selection strategy may be used even with the grid subject to large clustering ratios. The 2D weighted least squares $P2$ stencils used in this work correspond to a fixed number of $n = 13$ nearest neighbors (including the central node) and so are only 2.6 times the size of a finite difference Laplacian stencil. More importantly, these meshfree stencils result in consistent truncation errors as they enforce the polynomial reproducing conditions to the specified order.

To verify and validate different aspects of the solver, several benchmarks were presented on fixed and moving meshfree grids. In Chapter 3, two model equations were studied, namely the steady and transient heat equation. We verified the consistency of the weighted least squares approximation using second order and fourth order stencils on simple and complex domains. A detailed truncation error analysis was carried out to corroborate the reported numerical results. Notably, we also demonstrated how the local truncation error can be used to obtain accurate error estimates. Two other verification benchmarks were used to measure the convergence behavior when adaptivity is used to resolve a discontinuity and when adaptivity is used to resolve a smooth solution variation confined to a thin region of a 3D domain. Chapter 3 concluded with two validation benchmarks for incompressible flow, namely uniform flow over a cylinder and the classic lid driven cavity. For these steady flows, fixed meshfree grids were used with large clustering ratios to resolve the viscous flow near the boundaries. Varying the Reynolds numbers, good agreement was observed for various flow parameters as compared to literature values.

In Chapter 4, the semi-implicit approximation projection method was verified and validated while using fixed and moving quasi-uniform meshfree grids. The splitting error and convergence behavior was verified for the Taylor-Green vortex decay problem and modified lid driven cavity. For validation, the inline oscillating cylinder was studied at different Reynolds numbers. The resulting time periodic flow was validated against the weakly compressible lattice Boltzmann method and showed excellent agreement.

Lastly, in Chapter 5, three validation benchmarks with prescribed rigid body motion were studied while using grid adaptation near relevant boundaries. Namely, the inline oscillating cylinder in a small and large domain, and the two moving cylinder problem. Revisiting the inline oscillating cylinder, we showed that for a given number of grid nodes, the adaptive solution more accurately resolves the velocity profile when the Reynolds number is sufficiently high. Increasing the domain size by an order of magnitude, we then validated the drag coefficient for the inline oscillating cylinder against experimental and numerical results. In the final benchmark, the drag and lift coefficient of two cylinders narrowly passing each other in a large domain was validated against an immersed interface solver from the literature showing good agreement.

6.2 Future work

With respect to future work the following topics may be of interest:

1. **hp adaptivity**. We demonstrated distanced-based adaptivity in this work while fixing the local approximation to reproduce $P2$ polynomials. A more effective approach would be solution-based adaptivity. For example, one may use estimates of the local truncation error to 1) specify the spring equilibrium length and thereby control the grid spacing or 2) use the estimates to increase the local polynomial reproducing conditions e.g., from $P2$ to $P4$. Increasing the resolution in areas where the local truncation error is high will reduce the truncation error which in turn will decrease the solution error. Similarly, increasing the local polynomial reproducing conditions would remove high order local truncation errors and improve the solution. The approaches can be applied simultaneously and should be done in a manner such that the average local truncation error decreases overall according to the desired convergence rate. Exploring this topic first on fixed meshfree grids and then on moving meshfree grids may be worthwhile.

2. **Fluid-structure interaction**. The benchmarks detailed in this thesis all used prescribed rigid body motion. This is the simplest case of fluid-structure interaction where only the fluid interacts with the solid. Having showed we can accurately compute the forces for this simpler case, naturally the next stop would be to free the degrees of freedom of the solid and thereby couple the fluid forces to the rigid body equations of motion. More generally, the solid may not be rigid, and a set of governing equations describing the deformation of the solid material could be introduced.

3. **Stabilization of the convective term**. In all benchmarks the convective term was discretized using a GFD equivalent of central differences. For high Reynolds number flow, it is well known that this can lead to nonphysical oscillations. To handle higher Reynolds number flows, stabilization techniques needs to be applied. One common approach is through applying higher order upwinding to certain regions of the domain. Similarly, these studies should first be carried out on fixed meshfree grids for model equations and then later extended to moving meshfree grids.

4. **Deferred corrections**. Motivated by classic finite differences, we showed for the 2D Poisson equation, higher order solutions can similarly be obtained by applying deferred corrections to lower order meshfree GFD discretizations. This required solving a secondary problem for the error estimate and adding it to the lower order solution to improve its accuracy. The advantages of this approach are that 1) smaller stencils can be used to form the linear systems and 2) accurate error estimates are obtained which can be used to decide if refinement should continue or not. Investigating this approach further and applying it to more challenging problems may be worth while.

5. **Conservation**. As detailed in the opening paragraphs of this thesis, developing a meshfree discretization that simultaneously enforces the polynomial reproducing conditions and conservation is non-trivial. While groups have made progress, how to

do this cheaply and effectively for viscous incompressible flows on evolving mesh-free grids is still an open challenge.

Appendices

APPENDIX A

INLINE OSCILLATING CYLINDER

Table A.1: GFD solution for the oscillating cylinder horizontal velocity profile for $Re = 10$ and $Re = 100$. Data sampled using a quadratic WLS basis along the vertical line at $x = 1.0$. Approximately $N = 360$ nodes spanned the container side with Courant number $C_o \approx 0.14$. Columns correspond to time snapshots of the velocity profile, see Figure 4.7.

	$Re = 10$					$Re = 100$				
y	$6\pi/5$	$7\pi/5$	$8\pi/5$	$9\pi/5$	2π	$6\pi/5$	$7\pi/5$	$8\pi/5$	$9\pi/5$	2π
1.4525	0.5878	0.9511	0.9511	0.5878	0.0000	0.5878	0.9511	0.9511	0.5878	0.0000
1.4625	0.5878	0.9511	0.9511	0.5878	0.0321	0.5878	0.9511	0.9511	0.5878	0.1400
1.4725	0.5864	0.9511	0.9511	0.5857	0.0924	0.5814	0.9511	0.9511	0.5878	0.3943
1.4825	0.4098	0.9511	0.9511	0.5412	0.1391	0.1028	0.9511	0.9511	0.6469	0.5644
1.4925	0.2526	0.9511	0.9511	0.4810	0.1762	-0.2871	0.9511	0.9511	0.6251	0.6689
1.5025	0.1107	0.7680	0.8191	0.4147	0.2046	-0.5857	0.5482	0.7461	0.5425	0.7180
1.5125	-0.0171	0.5211	0.6228	0.3437	0.2248	-0.8064	0.0134	0.4001	0.4167	0.7223
1.5225	-0.1316	0.2955	0.4347	0.2689	0.2378	-0.9610	-0.4292	0.0545	0.2629	0.6909
1.5525	-0.4043	-0.2714	-0.0794	0.0310	0.2399	-1.1319	-1.2691	-0.8276	-0.2475	0.4636
1.5825	-0.5889	-0.6977	-0.5157	-0.2110	0.2009	-1.0366	-1.5585	-1.3375	-0.6610	0.1689
1.6125	-0.7058	-1.0098	-0.8737	-0.4400	0.1368	-0.8374	-1.5355	-1.5069	-0.8910	-0.0778
1.6425	-0.7723	-1.2297	-1.1558	-0.6433	0.0613	-0.6365	-1.3805	-1.4728	-0.9628	-0.2314
1.6725	-0.8028	-1.3759	-1.3662	-0.8120	-0.0145	-0.4845	-1.2024	-1.3584	-0.9441	-0.3002
1.7025	-0.8084	-1.4628	-1.5103	-0.9403	-0.0819	-0.3911	-1.0542	-1.2372	-0.8932	-0.3160
1.7325	-0.7974	-1.5017	-1.5936	-1.0247	-0.1344	-0.3441	-0.9522	-1.1416	-0.8437	-0.3087
1.7625	-0.7753	-1.5003	-1.6206	-1.0638	-0.1681	-0.3270	-0.8938	-1.0796	-0.8088	-0.2954
1.7925	-0.7449	-1.4630	-1.5947	-1.0573	-0.1811	-0.3301	-0.8699	-1.0478	-0.7878	-0.2800
1.8225	-0.7063	-1.3910	-1.5180	-1.0064	-0.1739	-0.3513	-0.8718	-1.0361	-0.7713	-0.2560
1.8525	-0.6572	-1.2828	-1.3907	-0.9133	-0.1489	-0.3920	-0.8895	-1.0288	-0.7423	-0.2111
1.8825	-0.5924	-1.1340	-1.2125	-0.7810	-0.1107	-0.4485	-0.9055	-1.0007	-0.6775	-0.1341
1.9125	-0.5044	-0.9379	-0.9819	-0.6144	-0.0662	-0.5007	-0.8864	-0.9140	-0.5528	-0.0251
1.9425	-0.3828	-0.6858	-0.6980	-0.4196	-0.0245	-0.5009	-0.7745	-0.7227	-0.3597	0.0885
1.9725	-0.2150	-0.3675	-0.3603	-0.2049	0.0031	-0.3627	-0.4889	-0.3944	-0.1343	0.1341
1.9825	-0.1460	-0.2448	-0.2359	-0.1307	0.0068	-0.2649	-0.3400	-0.2571	-0.0691	0.1132
1.9925	-0.0696	-0.1129	-0.1057	-0.0559	0.0067	-0.1317	-0.1603	-0.1113	-0.0193	0.0644
2.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Table A.2: GFD solution for the oscillating cylinder horizontal velocity profile for $Re = 400$ and $Re = 1000$. Data sampled using a quadratic WLS basis along the vertical line at $x = 1.0$. Approximately $N = 360$ nodes spanned the container side with Courant number $C_o \approx 0.14$. Columns correspond to time snapshots of the velocity profile, see Figure 4.8.

	$Re = 400$					$Re = 1000$				
y	$6\pi/5$	$7\pi/5$	$8\pi/5$	$9\pi/5$	2π	$6\pi/5$	$7\pi/5$	$8\pi/5$	$9\pi/5$	2π
1.4525	0.5878	0.9511	0.9511	0.5878	0.0000	0.5878	0.9511	0.9511	0.5878	0.0000
1.4575	0.5878	0.9511	0.9511	0.5878	0.0102	0.5878	0.9511	0.9511	0.5878	0.0183
1.4625	0.5878	0.9511	0.9511	0.5878	0.2779	0.5878	0.9511	0.9511	0.5878	0.4328
1.4675	0.5878	0.9511	0.9511	0.5878	0.5167	0.5878	0.9511	0.9511	0.5878	0.7393
1.4725	0.5735	0.9511	0.9511	0.5893	0.6789	0.5639	0.9511	0.9511	0.5906	0.8900
1.4775	0.1082	0.9511	0.9511	0.6692	0.7793	-0.1602	0.9511	0.9511	0.7048	0.9214
1.4825	-0.3101	0.9511	0.9511	0.6795	0.8304	-0.7160	0.9511	0.9511	0.6638	0.8772
1.4875	-0.6294	0.9511	0.9511	0.6246	0.8420	-1.0636	0.9511	0.9511	0.5030	0.7835
1.4925	-0.8679	0.9511	0.9511	0.5221	0.8212	-1.2569	0.9511	0.9511	0.2815	0.6601
1.4975	-1.0419	0.7748	0.8663	0.3899	0.7753	-1.3406	0.6697	0.8132	0.0413	0.5252
1.5025	-1.1607	0.1720	0.5433	0.2391	0.7109	-1.3465	-0.2387	0.2950	-0.1862	0.3922
1.5075	-1.2341	-0.3176	0.1997	0.0811	0.6337	-1.2994	-0.8759	-0.2259	-0.3808	0.2699
1.5125	-1.2699	-0.7176	-0.1338	-0.0753	0.5487	-1.2208	-1.3062	-0.6696	-0.5344	0.1644
1.5175	-1.2757	-1.0374	-0.4426	-0.2234	0.4604	-1.1270	-1.5585	-1.0176	-0.6482	0.0781
1.5225	-1.2578	-1.2801	-0.7153	-0.3573	0.3728	-1.0293	-1.6823	-1.2547	-0.7236	0.0107
1.5275	-1.2223	-1.4578	-0.9462	-0.4751	0.2885	-0.9358	-1.7135	-1.3974	-0.7698	-0.0393
1.5325	-1.1735	-1.5782	-1.1330	-0.5741	0.2103	-0.8521	-1.6884	-1.4659	-0.7942	-0.0749
1.5375	-1.1165	-1.6508	-1.2758	-0.6542	0.1392	-0.7797	-1.6313	-1.4835	-0.8042	-0.0991
1.5425	-1.0544	-1.6859	-1.3792	-0.7167	0.0770	-0.7200	-1.5611	-1.4716	-0.8055	-0.1150
1.5525	-0.9271	-1.6730	-1.4852	-0.7953	-0.0217	-0.6343	-1.4238	-1.4130	-0.7965	-0.1311
1.5825	-0.6347	-1.3917	-1.4042	-0.8198	-0.1498	-0.5325	-1.1932	-1.2698	-0.7610	-0.1392
1.6125	-0.5117	-1.1640	-1.2479	-0.7735	-0.1664	-0.4962	-1.0969	-1.1863	-0.7285	-0.1370
1.6425	-0.4661	-1.0502	-1.1557	-0.7422	-0.1681	-0.4718	-1.0273	-1.1162	-0.6965	-0.1316
1.6725	-0.4388	-0.9838	-1.0945	-0.7188	-0.1693	-0.4533	-0.9706	-1.0561	-0.6670	-0.1255
1.7025	-0.4177	-0.9332	-1.0442	-0.6962	-0.1682	-0.4386	-0.9238	-1.0049	-0.6406	-0.1196
1.7325	-0.4019	-0.8917	-1.0004	-0.6739	-0.1643	-0.4269	-0.8850	-0.9614	-0.6172	-0.1139
1.7625	-0.3908	-0.8581	-0.9625	-0.6524	-0.1582	-0.4177	-0.8530	-0.9247	-0.5965	-0.1083
1.7925	-0.3834	-0.8308	-0.9297	-0.6319	-0.1507	-0.4107	-0.8268	-0.8937	-0.5782	-0.1027
1.8225	-0.3777	-0.8083	-0.9016	-0.6137	-0.1437	-0.4060	-0.8056	-0.8678	-0.5619	-0.0966
1.8525	-0.3732	-0.7916	-0.8811	-0.6013	-0.1397	-0.4033	-0.7890	-0.8461	-0.5470	-0.0897
1.8825	-0.3779	-0.7904	-0.8766	-0.5990	-0.1357	-0.4017	-0.7754	-0.8275	-0.5330	-0.0825
1.9125	-0.4183	-0.8207	-0.8899	-0.5908	-0.1040	-0.4001	-0.7671	-0.8167	-0.5266	-0.0798
1.9425	-0.5095	-0.8569	-0.8602	-0.4988	0.0131	-0.4366	-0.7986	-0.8352	-0.5263	-0.0551
1.9575	-0.5435	-0.8261	-0.7709	-0.3819	0.1051	-0.4962	-0.8311	-0.8310	-0.4818	0.0129
1.9675	-0.5351	-0.7567	-0.6600	-0.2748	0.1621	-0.5388	-0.8287	-0.7820	-0.4000	0.0908
1.9775	-0.4766	-0.6255	-0.4989	-0.1558	0.1936	-0.5452	-0.7588	-0.6548	-0.2632	0.1755
1.9875	-0.3397	-0.4100	-0.2909	-0.0491	0.1705	-0.4444	-0.5531	-0.4159	-0.0961	0.2079
1.9975	-0.0824	-0.0927	-0.0585	-0.0010	0.0471	-0.1188	-0.1344	-0.0870	-0.0032	0.0669
2.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

REFERENCES

[1] T. Belytschko, Y. Y. Lu, and L. Gu, "Element-free galerkin methods," *International Journal for Numerical Methods in Engineering*, vol. 37, no. 2, pp. 229–256, 1994.

[2] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl, "Meshless methods: An overview and recent developments," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 3–47, 1996.

[3] O. V. Diyankov, "Uncertain grid method for numerical solution of pde," 2008.

[4] E. Kwan-yu Chiu, Q. Wang, R. Hu, and A. Jameson, "A conservative mesh-free scheme and generalized framework for conservation laws," *SIAM Journal on Scientific Computing*, vol. 34, no. 6, A2896–A2916, 2012.

[5] W. K. Liu, S. Jun, and Y. F. Zhang, "Reproducing kernel particle methods," *International Journal for Numerical Methods in Fluids*, vol. 20, no. 8-9, pp. 1081–1106, 1995.

[6] Y. Vasyliv and A. Alexeev, "Simulating incompressible flow on moving meshfree grids," *Computers & Fluids*, p. 104 464, Jan. 2020.

[7] W. Hu, G. Guo, X. Hu, D. Negrut, Z. Xu, and W. Pan, "A consistent spatially adaptive smoothed particle hydrodynamics method for fluid–structure interactions," *Computer Methods in Applied Mechanics and Engineering*, vol. 347, pp. 402–424, 2019.

[8] T Belytschko, Y Krongauz, J Dolbow, and C Gerlach, "On the completeness of meshfree particle methods," *International Journal for Numerical Methods in Engineering*, vol. 43, no. 5, pp. 785–819, 1998.

[9] S. N. Atluri and T. Zhu, "A new meshless local petrov-galerkin (mlpg) approach in computational mechanics," *Computational mechanics*, vol. 22, no. 2, pp. 117–127, 1998.

[10] T. Liszka and J Orkisz, "The finite difference method at arbitrary irregular grids and its application in applied mechanics," *Computers and Structures*, vol. 11, pp. 83–95, 1979.

[11] M. Liu and G. Liu, "Smoothed particle hydrodynamics (sph): An overview and recent developments," *Archives of computational methods in engineering*, vol. 17, no. 1, pp. 25–76, 2010.

[12] E. Onate, S. Idelsohn, O. C. Zienkiewicz, and R. L. Taylor, "A finite point method in computational mechanics. applications to convective transport and fluid flow," *International Journal for Numerical Methods in Engineering*, vol. 39, pp. 3839–3866, 22 1996.

[13] A. Katz and A. Jameson, "A meshless volume scheme," in *19th AIAA Computational Fluid Dynamics*, American Institute of Aeronautics and Astronautics, 2009.

[14] A. Colagrossi and M. Landrini, "Numerical simulation of interfacial flows by smoothed particle hydrodynamics," *Journal of Computational Physics*, vol. 191, no. 2, pp. 448–475, 2003.

[15] S. Marrone, M Antuono, A Colagrossi, G Colicchio, D Le Touzé, and G Graziani, "Delta-sph model for simulating violent impact flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 13, pp. 1526–1542, 2011.

[16] P. Omidvar, P. K. Stansby, and B. D. Rogers, "Sph for 3d floating bodies using variable mass particle distribution," *International Journal for Numerical Methods in Fluids*, vol. 72, no. 4, pp. 427–452, 2013.

[17] X. Hu and N. A. Adams, "An incompressible multi-phase sph method," *Journal of computational physics*, vol. 227, no. 1, pp. 264–278, 2007.

[18] X. Hu and N. Adams, "A constant-density approach for incompressible multi-phase sph," *Journal of Computational Physics*, vol. 228, no. 6, pp. 2082–2091, 2009.

[19] G. Oger, S. Marrone, D. L. Touzé, and M. de Leffe, "SPH accuracy improvement through the combination of a quasi-lagrangian shifting transport velocity and consistent ALE formalisms," *Journal of Computational Physics*, vol. 313, pp. 76–98, 2016.

[20] A. Khayyer and H. Gotoh, "Enhancement of performance and stability of MPS mesh-free particle method for multiphase flows characterized by high density ratios," *Journal of Computational Physics*, vol. 242, pp. 211–233, 2013.

[21] A. M. Tartakovsky, N. Trask, K. Pan, B. Jones, W. Pan, and J. R. Williams, "Smoothed particle hydrodynamics and its applications for multiphase flow and reactive transport in porous media," *Computational Geosciences*, vol. 20, no. 4, pp. 807–834, 2015.

[22] T. Belytschko, D. Organ, and C. Gerlach, "Element-free galerkin methods for dynamic fracture in concrete," *Computer Methods in Applied Mechanics and Engineering*, vol. 187, no. 3-4, pp. 385–399, 2000.

[23] T. Rabczuk and T. Belytschko, "Cracking particles: A simplified meshfree method for arbitrary evolving cracks," *International Journal for Numerical Methods in Engineering*, vol. 61, no. 13, pp. 2316–2343, 2004.

[24] C. S. Chew, K. Yeo, and C. Shu, "A generalized finite-difference (gfd) ale scheme for incompressible flows around moving solid bodies on hybrid meshfree–cartesian grids," *Journal of Computational Physics*, vol. 218, no. 2, pp. 510–548, 2006.

[25] W. Hu, N. Trask, X. Hu, and W. Pan, "A spatially adaptive high-order meshless method for fluid–structure interactions," *Computer Methods in Applied Mechanics and Engineering*, vol. 355, pp. 67–93, 2019.

[26] K. Y. Chiu, *A conservative meshless framework for conservation laws with applications in computational fluid dynamics*. Stanford University, 2011.

[27] N. Trask, P. Bochev, and M. Perego, "A conservative, consistent, and scalable meshfree mimetic method," Mar. 11, 2019. arXiv: `http://arxiv.org/abs/1903.04621v1 [math.NA]`.

[28] H. H. Hu, N. A. Patankar, and M. Zhu, "Direct numerical simulations of fluid–solid systems using the arbitrary lagrangian–eulerian technique," *Journal of Computational Physics*, vol. 169, no. 2, pp. 427–462, 2001.

[29] B. Perot and R. Nallapati, "A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows," *Journal of Computational Physics*, vol. 184, no. 1, pp. 192–214, 2003.

[30] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics: Theory and application to non-spherical stars," *Monthly Notices of the Royal Astronomical Society*, vol. 181, no. 3, pp. 375–389, 1977.

[31] S. Li and W. K. Liu, "Meshfree and particle methods and their applications," *Applied Mechanics Reviews*, vol. 55, no. 1, pp. 1–34, 2002.

[32] R Fatehi and M. Manzari, "Error estimation in smoothed particle hydrodynamics and a new scheme for second derivatives," *Computers & Mathematics with Applications*, vol. 61, no. 2, pp. 482–498, 2011.

[33] J. P. Morris, P. J. Fox, and Y. Zhu, "Modeling low reynolds number incompressible flows using sph," *Journal of computational physics*, vol. 136, no. 1, pp. 214–226, 1997.

[34] E.-S. Lee, C. Moulinec, R. Xu, D. Violeau, D. Laurence, and P. Stansby, "Comparisons of weakly compressible and truly incompressible algorithms for the SPH

mesh free particle method," *Journal of computational physics*, vol. 227, no. 18, pp. 8417–8436, 2008.

[35] W. Hu, W. Pan, M. Rakhsha, Q. Tian, H. Hu, and D. Negrut, "A consistent multi-resolution smoothed particle hydrodynamics method," *Computer Methods in Applied Mechanics and Engineering*, vol. 324, pp. 278–299, 2017.

[36] S. J. Cummins and M. Rudman, "An sph projection method," *Journal of computational physics*, vol. 152, no. 2, pp. 584–607, 1999.

[37] R. Xu, P. Stansby, and D. Laurence, "Accuracy and stability in incompressible sph (isph) based on the projection method and a new approach," *Journal of computational Physics*, vol. 228, no. 18, pp. 6703–6725, 2009.

[38] S. Khorasanizade and J. M. Sousa, "A detailed study of lid-driven cavity flow at moderate reynolds numbers using incompressible sph," *International Journal for Numerical Methods in Fluids*, vol. 76, no. 10, pp. 653–668, 2014.

[39] C. W. Hirt, A. A. Amsden, and J. Cook, "An arbitrary lagrangian-eulerian computing method for all flow speeds," *Journal of computational physics*, vol. 14, no. 3, pp. 227–253, 1974.

[40] A. Chaniotis, D Poulikakos, and P Koumoutsakos, "Remeshed smoothed particle hydrodynamics for the simulation of viscous and heat conducting flows," *Journal of Computational Physics*, vol. 182, no. 1, pp. 67–90, 2002.

[41] G. Oger, M. Doring, B. Alessandrini, and P. Ferrant, "Two-dimensional sph simulations of wedge water entries," *Journal of computational physics*, vol. 213, no. 2, pp. 803–822, 2006.

[42] J Feldman and J Bonet, "Dynamic refinement and boundary contact forces in sph with applications in fluid flow problems," *International Journal for Numerical Methods in Engineering*, vol. 72, no. 3, pp. 295–324, 2007.

[43] R Vacondio, B. Rogers, P. Stansby, P Mignosa, and J Feldman, "Variable resolution for sph: A dynamic particle coalescing and splitting scheme," *Computer Methods in Applied Mechanics and Engineering*, vol. 256, pp. 132–148, 2013.

[44] D. A. Barcarolo, D. Le Touzé, G. Oger, and F. De Vuyst, "Adaptive particle refinement and derefinement applied to the smoothed particle hydrodynamics method," *Journal of Computational Physics*, vol. 273, pp. 640–657, 2014.

[45] X. Bian, Z. Li, and G. E. Karniadakis, "Multi-resolution flow simulations by smoothed particle hydrodynamics via domain decomposition," *Journal of Computational Physics*, vol. 297, pp. 132–155, 2015.

[46] B. Nayroles, G. Touzot, and P. Villon, "Generalizing the finite element method: Diffuse approximation and diffuse elements," *Computational Mechanics*, vol. 10, pp. 307–318, 5 1992.

[47] P. Lancaster and K. Salkauskas, "Surfaces generated by moving least squares methods," *Math. Comp.*, vol. 37, no. 155, pp. 141–158, 1981.

[48] S. Fernández-Méndez and A. Huerta, "Imposing essential boundary conditions in mesh-free methods," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 12-14, pp. 1257–1275, 2004.

[49] J. Dolbow and T. Belytschko, "Numerical integration of the galerkin weak form in meshfree methods," *Computational Mechanics*, vol. 23, no. 3, pp. 219–230, 1999.

[50] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Numerical Mathematics and Scie, 2014.

[51] S. Beissel and T. Belytschko, "Nodal integration of the element-free galerkin method," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 49–74, 1996.

[52] A. Huerta, Y. Vidal, and P. Villon, "Pseudo-divergence-free element free galerkin method for incompressible fluid flow," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 12-14, pp. 1119–1136, 2004.

[53] L. Cueto-Felgueroso, I. Colominas, X. Nogueira, F. Navarrina, and M. Casteleiro, "Finite volume solvers and moving least-squares approximations for the compressible navier–stokes equations on unstructured grids," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 45-48, pp. 4712–4736, 2007.

[54] L. Ramírez, X. Nogueira, S. Khelladi, J.-C. Chassaing, and I. Colominas, "A new higher-order finite volume method based on moving least squares for the resolution of the incompressible navier–stokes equations on unstructured grids," *Computer Methods in Applied Mechanics and Engineering*, vol. 278, pp. 883–901, 2014.

[55] W. K. Liu and Y. Chen, "Wavelet and multiple scale reproducing kernel methods," *International Journal for Numerical Methods in Fluids*, vol. 21, no. 10, pp. 901–931, 1995.

[56] J.-S. Chen, C. Pan, C.-T. Wu, and W. K. Liu, "Reproducing kernel particle methods for large deformation analysis of non-linear structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 195–227, 1996.

[57] N. R. Aluru, "A point collocation method based on reproducing kernel approximations," *International Journal for Numerical Methods in Engineering*, vol. 47, no. 6, pp. 1083–1121, 2000.

[58] W. K. Liu, Y. Chen, S. Jun, J. S. Chen, T. Belytschko, C. Pan, R. A. Uras, and C. T. Chang, "Overview and applications of the reproducing kernel particle methods," *Archives of Computational Methods in Engineering*, vol. 3, no. 1, pp. 3–80, 1996.

[59] E. Oñate, S. Idelsohn, O. Zienkiewicz, R. Taylor, and C. Sacco, "A stabilized finite point method for analysis of fluid mechanics problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 315–346, 1996.

[60] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

[61] J. Batina, "A gridless euler/navier-stokes solution algorithm for complex-aircraft applications," in *31st Aerospace Sciences Meeting*, American Institute of Aeronautics and Astronautics, 1993.

[62] N. Perrone and R. Kao, "A general finite difference method for arbitrary meshes," *Computers & Structures*, vol. 5, no. 1, pp. 45–57, 1975.

[63] N. Aluru and G. Li, "Finite cloud method: A true meshless technique based on a fixed reproducing kernel approximation," *International Journal for Numerical Methods in Engineering*, vol. 50, no. 10, pp. 2373–2410, 2001.

[64] Y. Luo and U. Hussler-Combe, "A generalized finite-difference method based on minimizing global residual," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 13-14, pp. 1421–1438, 2002.

[65] L. Gavete, M. Gavete, and J. Benito, "Improvements of generalized finite difference method and comparison with other meshless method," *Applied Mathematical Modelling*, vol. 27, no. 10, pp. 831–847, 2003.

[66] H. Ding, C. Shu, K. Yeo, and D. Xu, "Simulation of incompressible viscous flows past a circular cylinder by hybrid FD scheme and meshless least square-based finite difference method," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 9-11, pp. 727–744, 2004.

[67] B. Mendez and A. Velazquez, "Finite point solver for the simulation of 2-d laminar incompressible unsteady flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 9-11, pp. 825–848, 2004.

[68] S. Tiwari and J. Kuhnert, "Finite pointset method based on the projection method for simulations of the incompressible navier-stokes equations," *Meshfree methods for partial differential equations*, pp. 373–387, 2002.

[69] X. Jin, G. Li, and N. Aluru, "Positivity conditions in meshless collocation methods," *Computer methods in applied mechanics and engineering*, vol. 193, no. 12, pp. 1171–1202, 2004.

[70] X. Wang, K. Yeo, C. Chew, and B. Khoo, "A SVD-GFD scheme for computing 3d incompressible viscous fluid flows," *Computers & Fluids*, vol. 37, no. 6, pp. 733–746, 2008.

[71] L. Demkowicz, A. Karafiat, and T. Liszka, "On some convergence results for FDM with irregular mesh," *Computer Methods in Applied Mechanics and Engineering*, vol. 42, no. 3, pp. 343–355, 1984.

[72] S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*. CRC Press, 2018.

[73] C. A. Duarte and J. T. Oden, "H-p clouds—anh-p meshless method," *Numerical Methods for Partial Differential Equations*, vol. 12, no. 6, pp. 673–705, 1996.

[74] C. Duarte and J. Oden, "An h-p adaptive method using clouds," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 237–262, 1996.

[75] J. Melenk and I. Babuška, "The partition of unity finite element method: Basic theory and applications," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1-4, pp. 289–314, 1996.

[76] C. Duarte, I. Babuška, and J. Oden, "Generalized finite element methods for three-dimensional structural mechanics problems," *Computers & Structures*, vol. 77, no. 2, pp. 215–232, 2000.

[77] Y. L. Wu, G. R. Liu, and Y. T. Gu, "Application of meshless local petrov-galerkin (MLPG) approach to simulation of incompressible flow," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 48, no. 5, pp. 459–475, 2005.

[78] P.-O. Persson and G. Strang, "A simple mesh generator in MATLAB," *SIAM review*, vol. 46, no. 2, pp. 329–345, 2004.

[79] R. J. Leveque, *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*. CAMBRIDGE, Sep. 1, 2007, 341 pp., ISBN: 0898716292.

[80] G. E. Fasshauer, *Meshfree Approximation Methods with MATLAB [With CDROM]*. WORLD SCIENTIFIC PUB CO INC, Jun. 1, 2007, 500 pp., ISBN: 981270633X.

[81] R. Brun and F. Rademakers, "Rootan object oriented data analysis framework," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators,*

Spectrometers, Detectors and Associated Equipment, vol. 389, no. 1-2, pp. 81–86, 1997.

[82] G. Guennebaud, B. Jacob, *et al.*, *Eigen v3*, http://eigen.tuxfamily.org, 2010.

[83] T. J. Hughes, *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[84] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical mathematics*. Springer Science & Business Media, 2010, vol. 37.

[85] J. B. Perot, "An analysis of the fractional step method," *Journal of Computational Physics*, vol. 108, no. 1, pp. 51–58, 1993.

[86] J. K. Dukowicz and A. S. Dvinsky, "Approximate factorization as a high order splitting for the implicit incompressible flow equations," *Journal of Computational Physics*, vol. 102, no. 2, pp. 336–347, 1992.

[87] A. Quarteroni, F. Saleri, and A. Veneziani, "Factorization methods for the numerical approximation of navier–stokes equations," *Computer methods in applied mechanics and engineering*, vol. 188, no. 1-3, pp. 505–526, 2000.

[88] M. O. Henriksen and J. Holmen, "Algebraic splitting for incompressible navier–stokes equations," *Journal of Computational Physics*, vol. 175, no. 2, pp. 438–453, 2002.

[89] F. Saleri and A. Veneziani, "Pressure correction algebraic splitting methods for the incompressible navier–stokes equations," *SIAM journal on numerical analysis*, vol. 43, no. 1, pp. 174–194, 2005.

[90] J Donea, A. Huerta, J.-P. Ponthot, and A Rodriguez-Ferran, *Encyclopedia of computational mechanics vol. 1: Fundamentals., chapter 14: Arbitrary lagrangian-eulerian methods*, 2004.

[91] A. J. Barlow, P.-H. Maire, W. J. Rider, R. N. Rieben, and M. J. Shashkov, "Arbitrary lagrangian–eulerian methods for modeling high-speed compressible multimaterial flows," *Journal of Computational Physics*, vol. 322, pp. 603–665, 2016.

[92] P.-O. Persson, "Mesh generation for implicit geometries," PhD thesis, Massachusetts Institute of Technology, 2005.

[93] Z. Li, "A fast iterative algorithm for elliptic interface problems," *SIAM Journal on Numerical Analysis*, vol. 35, no. 1, pp. 230–254, 1998.

[94] J. B. Perot and V Subramanian, "Discrete calculus methods for diffusion," *Journal of Computational Physics*, vol. 224, no. 1, pp. 59–81, 2007.

[95] Y. Vasyliv and A. Alexeev, "Development of general finite differences for complex geometries using a sharp interface formulation," *Computers & Fluids*, vol. 193, p. 103 959, 2019.

[96] U. Ghia, K. N. Ghia, and C. Shin, "High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method," *Journal of computational physics*, vol. 48, no. 3, pp. 387–411, 1982.

[97] T. Ye, R. Mittal, H. Udaykumar, and W. Shyy, "An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries," *Journal of computational physics*, vol. 156, no. 2, pp. 209–240, 1999.

[98] S. Dennis and G.-Z. Chang, "Numerical solutions for steady flow past a circular cylinder at reynolds numbers up to 100," *J. Fluid Mech*, vol. 42, no. 3, pp. 471–489, 1970.

[99] S. Etienne, A. Garon, and D. Pelletier, "Code verification for unsteady flow simulations with high order time-stepping schemes," in *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, 2009, p. 169.

[100] O Botella and R Peyret, "Benchmark spectral results on the lid-driven cavity flow," *Computers & Fluids*, vol. 27, no. 4, pp. 421–433, 1998.

[101] W. L. Oberkampf and T. G. Trucano, "Verification and validation in computational fluid dynamics," *Progress in Aerospace Sciences*, vol. 38, no. 3, pp. 209–272, 2002.

[102] T. Shih, C. Tan, and B. Hwang, "Effects of grid staggering on numerical schemes," *International Journal for numerical methods in fluids*, vol. 9, no. 2, pp. 193–212, 1989.

[103] C.-C. Liao, Y.-W. Chang, C.-A. Lin, and J. McDonough, "Simulating flows with moving rigid boundary using immersed-boundary method," *Computers & Fluids*, vol. 39, no. 1, pp. 152–167, 2010.

[104] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E. M. Viggen, *The Lattice Boltzmann Method: Principles and Practice*. Springer, 2016.

[105] X. He and L.-S. Luo, "A priori derivation of the lattice boltzmann equation," *Physical Review E*, vol. 55, no. 6, R6333, 1997.

[106] X. Shan, X.-F. Yuan, and H. Chen, "Kinetic theory representation of hydrodynamics: A way beyond the navier-stokes equation," *Journal of Fluid Mechanics*, vol. 550, no. 1, p. 413, 2006.

[107] M. Bouzidi, M. Firdaouss, and P. Lallemand, "Momentum transfer of a boltzmann-lattice fluid with boundaries," *Physics of Fluids (1994-present)*, vol. 13, no. 11, pp. 3452–3459, 2001.

[108] J. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics*. Springer Berlin Heidelberg, 2001, ISBN: 9783540420743.

[109] P.-O. Persson, "Mesh size functions for implicit geometries and pde-based gradient limiting," *Engineering with Computers*, vol. 22, no. 2, pp. 95–109, 2006.

[110] H. DTSCH, F. DURST, S. BECKER, and H. LIENHART, "Low-reynolds-number flow around an oscillating circular cylinder at low keulegan–carpenter numbers," *Journal of Fluid Mechanics*, vol. 360, pp. 249–271, 1998.

[111] D. Kim and H. Choi, "Immersed boundary method for flow around an arbitrarily moving body," *Journal of Computational Physics*, vol. 212, no. 2, pp. 662–680, 2006.

[112] D. Russell and Z. J. Wang, "A cartesian grid method for modeling multiple moving objects in 2d incompressible viscous flow," *Journal of Computational Physics*, vol. 191, no. 1, pp. 177–205, 2003.

[113] S. Xu and Z. J. Wang, "An immersed interface method for simulating the interaction of a fluid with moving boundaries," *Journal of Computational Physics*, vol. 216, no. 2, pp. 454–493, 2006.